Kementerian Pendidikan Dan Kebudayaan
Republik Indonesia
2013



PEREKAYASAAN SISTEM CONTROL

UNTUK SMA/MAK KELAS X

SEMESTER 1



HAK CIPTA

Penulis	:	**ARIE ERIC RAWUNG**
Editor Materi	:	
Editor Bahasa	:	
Ilustrasi Sampul	:	
Desain & Ilustrasi Buku	:	**PPPPTK BOE Malang**

Pusat Pengembangan & Pemberdayaan Pendidik & Tenaga Kependidikan Bidang Otomotif & Elektronika:

Jl. Teluk Mandar, Arjosari Tromol Pos 5, Malang 65102, Telp. (0341) 491239, (0341) 495849, Fax. (0341) 491342, Surel: vedcmalang@vedcmalang.or.id, Laman: www.vedcmalang.com



DISKLAIMER (*DISCLAIMER*)

Penerbit tidak menjamin kebenaran dan keakuratan isi/informasi yang tertulis di dalam buku tek ini. Kebenaran dan keakuratan isi/informasi merupakan tanggung jawab dan wewenang dari penulis.

Penerbit tidak bertanggung jawab dan tidak melayani terhadap semua komentar apapun yang ada didalam buku teks ini. Setiap komentar yang tercantum untuk tujuan perbaikan isi adalah tanggung jawab dari masing-masing penulis.

Setiap kutipan yang ada di dalam buku teks akan dicantumkan sumbernya dan penerbit tidak bertanggung jawab terhadap isi dari kutipan tersebut. Kebenaran keakuratan isi kutipan tetap menjadi tanggung jawab dan hak diberikan pada penulis dan pemilik asli. Penulis bertanggung jawab penuh terhadap setiap perawatan (perbaikan) dalam menyusun informasi dan bahan dalam buku teks ini.

Kewenangan Penerbit hanya sebatas memindahkan atau menerbitkan mempublikasi, mencetak, memegang dan memproses data sesuai dengan undang-undang yang berkaitan dengan perlindungan data.

Katalog Dalam Terbitan (KDT)

Perekayasaan Sistem Kontrol 2013

Kementerian Pendidikan & Kebudayaan

Direktorat Jenderal Peningkatan Mutu Pendidik & Tenaga Kependidikan, th. 2013: Jakarta



KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan yang Maha Esa atas tersusunnya buku teks ini, dengan harapan dapat digunakan sebagai buku teks untuk siswa Sekolah Menengah Kejuruan (SMK) Bidang Studi Keahlian Teknologi dan Rekayasa, Program Keahlian Perekayasaan Sistem Kontrol..

Penerapan kurikulum 2013 mengacu pada paradigma belajar kurikulum abad 21 menyebabkan terjadinya perubahan, yakni dari pengajaran (*teaching*) menjadi BELAJAR (*learning*), dari pembelajaran yang berpusat kepada guru (*teachers-centered*) menjadi pembelajaran yang berpusat kepada peserta didik (*student-centered*), dari pembelajaran pasif (*pasive learning*) ke cara belajar peserta didik aktif (*active learning-CBSA*) atau *Student Active Learning-SAL*.

Buku teks ″ PEREKAYASAAN SISTEM KONTROL KELAS X SEMESTER 1″ ini disusun berdasarkan tuntutan paradigma pengajaran dan pembelajaran kurikulum 2013 diselaraskan berdasarkan pendekatan model pembelajaran yang sesuai dengan kebutuhan belajar kurikulum abad 21, yaitu pendekatan model pembelajaran berbasis peningkatan keterampilan proses sains.

Penyajian buku teks untuk Mata Pelajaran ″PEREKAYASAAN SISTEM KONTROL” ini disusun dengan tujuan agar supaya peserta didik dapat melakukan proses pencarian pengetahuan berkenaan dengan materi pelajaran melalui berbagai aktivitas proses sains sebagaimana dilakukan oleh para ilmuwan dalam melakukan eksperimen ilmiah (penerapan scientifik), dengan demikian peserta didik diarahkan untuk menemukan sendiri berbagai fakta, membangun konsep, dan nilai-nilai baru secara mandiri.

Kementerian Pendidikan dan Kebudayaan, Direktorat Pembinaan Sekolah Menengah Kejuruan, dan Direktorat Jenderal Peningkatan Mutu Pendidik dan Tenaga Kependidikan menyampaikan terima kasih, sekaligus saran kritik demi kesempurnaan buku teks ini dan penghargaan kepada semua pihak yang telah berperan serta dalam membantu terselesaikannya buku teks siswa untuk Mata Pelajaran ″PEREKAYASAAN SISTEM KONTROL” kelas X Semester 1 Sekolah Menengah Kejuruan (SMK).

Jakarta, 12 Desember 2013

Menteri Pendidikan dan Kebudayaan

Prof. Dr. Mohammad Nuh, DEA



DAFTAR ISI













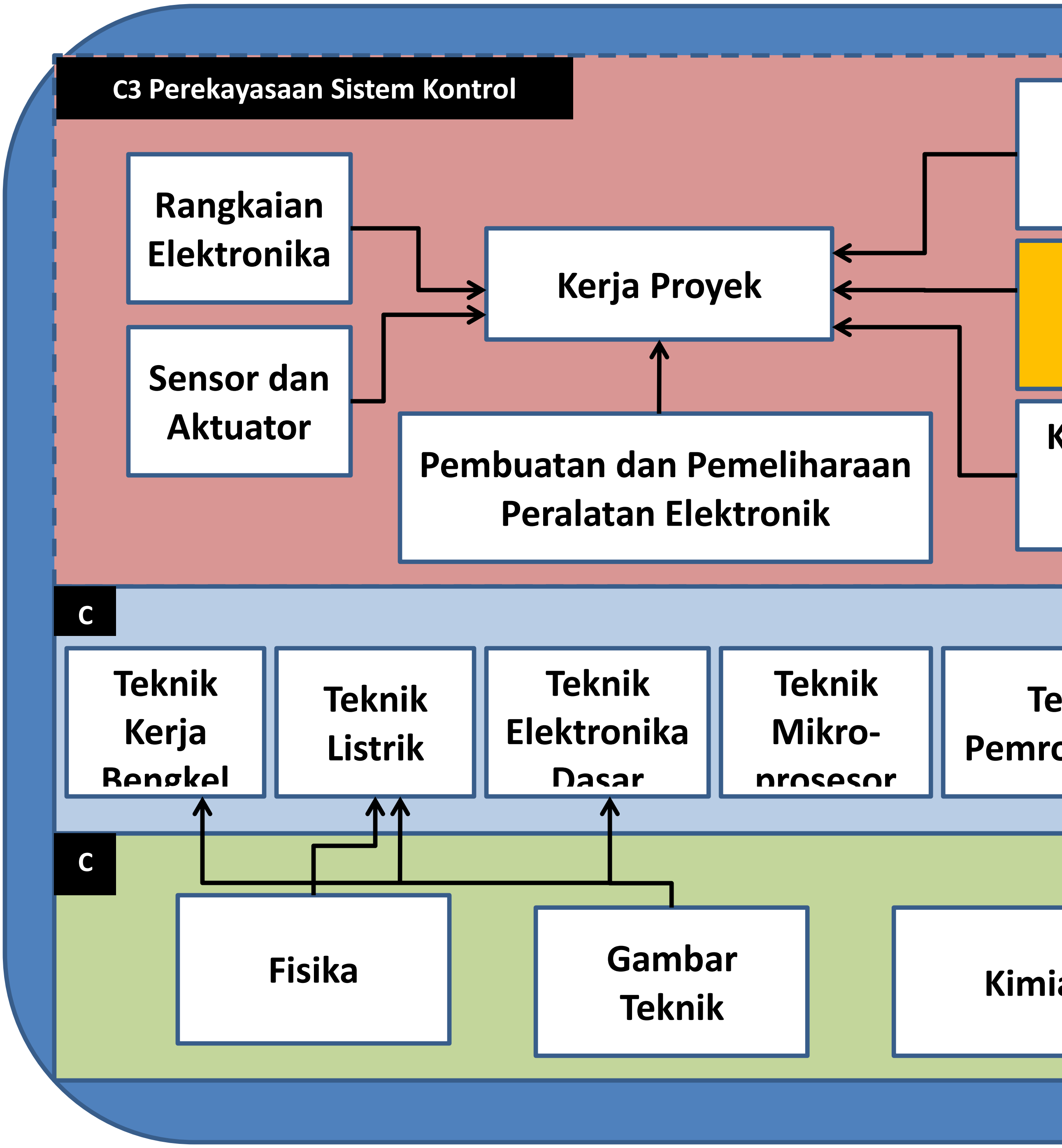








PETA KEDUDUKAN MODUL





A. DESKRIPSI MATERI PEMBELAJARAN

Rekayasa teknik kontrol berkaitan dengan pemahaman dan pengontrolan bahan dan kekuatan alam untuk kepentingan umat manusia . Tujuan modul rekayasa teknik kontrol ini untuk memberikan pemahaman kepada siswa mengenai dasar sistem kontrol di industri mensyaratkan bahwa sistem dapat dipahami dan dimodelkan untuk menghasilkan kontrol yang bersifat efektif. Tantangan saat ini untuk kontrolan adalah pemodelan dan kontrolan modern, kompleks, sistem yang saling terkait seperti sistem kontrol lalu lintas , proses kimia , dan sistem robot .

Pada buku siswa ini dibahas tentang dasar MATLAB untuk analisa dunia teknik, dasar sistem kontrol, Livewire sebagai alat bantu mensimulasikan desain rancangan elektronika, dasar Mikrokontroller dan Pengenalan PLC sebagai komponen utama dari suatu sistem untuk melakukan fungsi kontrol input-output.

.

B. PRASYARAT

Materi Rekayasa Teknik Kontrol 1 memberikan bekal awal dalam memahami kompetensi teknik kontrol pada jurusan teknik elektronika industri. Materi ini disampaikan pada kelas XI semester 1.

C. PETUNJUK PENGGUNAAN

Buku ini disusun dengan memberikan penjelasan tentang konsep dasar pemrograman MATLAB, dasar kontrol, simulator Livewire, mikrokontroller dan PLC dengan beberapa contoh aplikasi permodelan sederhana yang berkaitan dengan dunia teknik pada umumnya dan elektronika industri pada khususnya. Untuk memungkinkan siswa belajar sendiri secara tuntas , maka perlu diketahui bahwa isi buku ini pada setiap kegiatan belajar umumnya terdiri atas, uraian materi, contoh-contoh aplikasi, tugas dan tes formatif serta lembar kerja, sehingga diharapkan siswa dapat belajar mandiri (*individual learning*) dan *mastery learning* (belajar tuntas) dapat tercapai.



D. TUJUAN AKHIR

Tujuan akhir yang hendak dicapai adalah agar siswa mampu:

- Memahami terminologi sistem kontrol closed loop dan open loop serta menganalis komponen perkomponen dengan melakukan permodelan dengan menggunakan MATLAB
- Memahami simulasi karakteristik *transient response* system kontrol dari contoh-contoh dasar komponen elektronik dan mekanik yang disajikan didalam buku siswa.
- Mengenal komponen-komponen aktuator yang sering digunakan dalam dunia kontrol
- Mengenal dan memahami pemrograman mikrokontroller dengan menggunakan bahasa basic.
- Mengenal dasar dan pemrograman PLC serta aplikasinya untuk melakukan kontrol ON-OFF.



KOMPETENSI INTI (KI-3)	KOMPETENSI DASAR (KI-4)
Kompetensi Dasar (KD) : Memahamai prinsip dasar sistem control	**Kompetensi Dasar (KD) :** Mengenal dasar pemrograman Mikrokontroller
Indikator : - Memahami terminologi dan Simbol (perbandingan system *open-loop* versus *closed- loop*) - Mengenal software control dan electronic (*Matlab,danLivewire/ EWB/National Instruments/ Eagle*) - Memahami jenis desain sistem (Continues : Analog & Diskrit:Digital) - Mampu menerapkan simulasi karakteristik *transient response* systemdengan menggunakan MATLAB	**Indikator :** - Memahami perbedaan mikroprosesor vs. mikrokontroler - Mengenal Arsitektur Mikrokontroler AVR. - Memahami Fungsi masing-masing blok Mikrokontroler (memori, clock CPU, register, timer, counter, I/O, dll.) - Mengenal Instruksi , Flow chart Pemrograman pada Mikrokontroler (dengan bahasa Basic). - Mampu mensimulasikan dan program Mikrokontroler (operasi aritmathik, logika, baca/tulis, panggil, loncat, interupsi, Input/output dll).
Kompetensi Dasar (KD) : Mengenal dasar karakteristik Transient Response	**Kompetensi Dasar (KD) :** Mengenal dasar pemrograman PLC
Indikator : - Memahami dasar-dasar bentuk signal respon(step, dan impulse) - Mengenal karakteristik sistem ber-orde (orde satu, dan orde dua) - Simulasi Penerapan software "Matlab" atau software yang lain (untuk simulasi karakteristik *transient response*system) - Memahami penerapan Proses pengukuran besaran signal control analog dan/digital.	**Indikator :** - Mengenal sejarah perkembangan PLC, dan perbandingan kontroler PLC dengan Relay - Konsep Dasar PLC (Blok diagram, Simbol operasi , Prinsip kerja dan Fungsi) - Pemahaman instruksi dan fungsi blok ysng penting PLC (latch; timer; counter; MCR; fungsi logika, dan algorithma) - Pengenalan Bahasa pemrograman/ instruksi pada PLC serta software secara umum untuk operasi kontrol.



KEGIATAN BELAJAR 1

Sebelum proses pembelajaran di kelas berlangsung, sebaiknya siswa mempersiapkan diri dengan belajar mandiri sesuai dengan urutan materi yang akan diberikan.Sebagai gambaran kegiatan belajar siswa seperti pada tabel berikut :

NO	KEGIATAN SISWA	KETERANGAN
1	Persiapan Kegiatan 1 1. Siswa membaca materi pendahuluan 2. Siswa mempelajari pengenalan software MATLAB 3. Siswa mempelajari variabel, perintah logika dan aritmatika pada MATLAB 4. Siswa mencoba mengerjakan soal tes formatif 1	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 1
2	Persiapan Kegiatan 2 1. Siswa membaca materi pendahuluan 2. Siswa mempelajari materi Matriks dan array pada MATLAB 3. Siswa mempelajari perintah operasi array pada MATLAB 4. Siswa mencoba mengerjakan soal tes formatif 2	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 2
3	Persiapan Kegiatan 3 1. Siswa mempelajari materi Pendahuluan 2. Siswa mempelajari Menggambar grafik pada MATLAB 3. Siswa mencoba mengerjakan soal tes formatif 3	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 3



Selanjutnya siswa mendengarkan penyampaian materi pembelajaran di setiap pertemuan oleh guru serta menyesuaikan dengan model pembelajaran yang digunakan. Misalnya saatnya harus aktif mengerjakan soal maupun praktikum, maka siswa juga harus aktif dan kreatif. Melalui langkah kegiatan pembelajaran yang saling melengkapi diharapkan siswa dapat mencapai kompetensi yang distandarkan.

A. Tujuan Pembelajaran

Setelah mempelajari materi tentang dasar teknik kontrol, diharapkan siswa dapat:

1. mengidentifikasi ....
2. mengidentifikasi ....

B. Uraian Materi

- Dasar sistem kendali Mikrokontroller, komponen dan spesifikasinya serta perbandingan sistem kendali Mikrokontroller dengan sistem kendali yang lain.
- Teknik pemrograman Mikrokontroller.
- Teknik pemasangan dan pengawatan peralatan input output.
- Penggunaan alat pemrogram dengan komputer yang dilengkapi dengan software ladder
- Pengoperasian sistem kendali Mikrokontroller

C. Alokasi Waktu

4 jam pelajaran

D. Metode Pembelajaran

Teori dan Praktek

E. Media pembelajaran

- PC/Notebook
- Windows 7
- Livewire



KEGIATAN 1

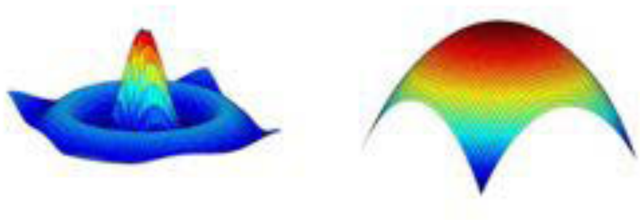
1.1 Pengertian Matlab

MATLAB merupakan suatu program komputer yang bisa membantu memecahkan berbagai masalah matematis yang kerap kita temui dalam bidang teknis. Kita bisa memanfaatkan kemampuan MATLAB untuk menemukan solusi dari berbagai masalah numerik secara cepat, mulai hal yang paling dasar, misalkan sistem 2 persamaan dengan 2 variabel:

$x - 2y = 32$

$12x + 5y = 12$

hingga yang kompleks, seperti mencari akar-akar polinomial, interpolasi dari sejumlah data, perhitungan dengan matriks, pengolahan sinyal, dan metoda numerik. Salah satu aspek yang sangat berguna dari MATLAB ialah kemampuannya untuk menggambarkan berbagai jenis grafik, sehingga kita bisa memvisualisasikan data dan fungsi yang kompleks. Sebagai contoh, tiga gambar berikut diciptakan dengan perintah surf di MATLAB.

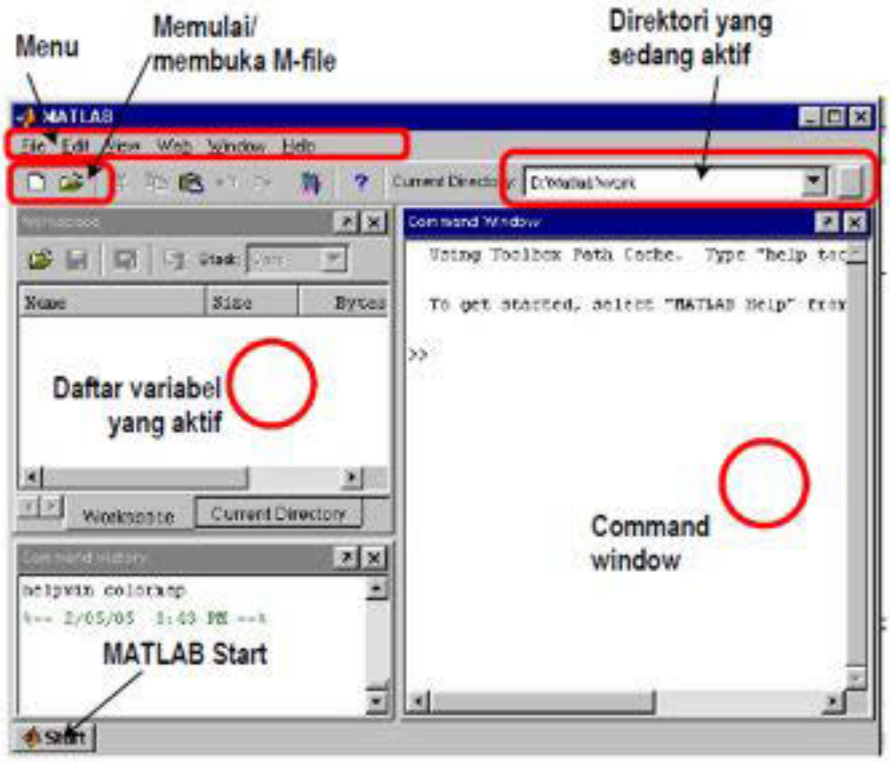


Gambar 1. 1 Grafik 3-dimensi dengan perintah “surf” di MATLAB.

Dalam buku ini kita akan mempelajari MATLAB setahap demi setahap, mulai dari hal yang sederhana hingga yang cukup kompleks. Yang perlu kita persiapkan untuk belajar MATLAB ialah seperangkat komputer yang sudah terinstal program MATLAB di dalamnya. Kita bisa gunakan MATLAB versi 5, 6 ataupun 7 untuk mempraktekkan berbagai contoh yang ada di buku ini. Di dalam buku ini kita akan mempelajari ‘teori’ penggunaan MATLAB, namun untukmenjadi mahir Anda harus duduk di depan komputer dan mempraktekkannya secara langsung!

1.2 Memulai MATLAB

Kita memulai MATLAB dengan mengeksekusi ikon MATLAB di layar komputer ataupun melalui tombol Start di Windows.Setelah proses loading program, jendela utama MATLAB akanmuncul seperti berikut ini.



Gambar 1. 2Jendela utama MATLAB.

Setelah proses loading usai, akan muncul prompt perintah didalam jendela perintah:

```
>>
```

Dari prompt inilah kita bisa mengetikkan berbagai perintahMATLAB, seperti halnya prompt di dalam DOS.



Sebagai permulaan, mari kita ketikkan perintah date :

```
>> date
```

setelah menekan Enter, akan muncul

```
>>ans =

            05-Feb-2005
```

date adalah perintah MATLAB untuk menampilkan tanggal hari ini. Berikutnya cobalah perintah clc untuk membersihkan jendela perintah:

```
>> clc
```

Ketika kita selesai dengan sesi MATLAB dan ingin keluar, gunakan perintah exit atau quit.

```
>> exit
```

atau...

```
>> quit
```

Atau bisa juga dengan menggunakan menu: File □ Exit MATLAB.

1.3 Sintaks Dasar Matlab

Jendela perintah window MATLAB berlaku seperti kalkulator komplek, dimana kita memberi sebuah perintah dan MATLAB mengeksekusi dengan benar.

Ketikan sebuah ekspresi yang valid, sebagai contoh :

```
>>5 + 5
```

Dan tekan ENTER, Bila tombol eksekusi ditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>ans = 10
```

Ketikan contoh berikut ini :

```
>>3^2               % 3 pangkat 2
```



Bila tombol eksekusiditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>ans = 9
```

Contoh lain :

```
>>sin(pi/2)                    % sin sudut 90o
```

Bila tombol eksekusiditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>ans = 1
```

Semicolon (;) menunjukkan akhir dari statemen.Dimaksud un tuk menampilkan atau menyembunyikan keluaran hasil MATLAB sebuah ekspresi, menambahkan sebuah semicolon pada akhir ekspresi.

Sebagai contoh :

```
>>x = 3;

>>y = x + 5
```

Bila tombol eksekusiditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>y = 8
```

Menambah sebuah symbol persen (%) digunakan untuk menunjukan sebuah komentar.

Sebagai contoh :

```
>>x = 9            % Memberikan nilai 9 ke variabel x
```

Dapat juga menuliskan sebuah blok komentar dengan menggunakan operator %{ and %}.



1.3.1 Operator dan Karakter Khusus

MATLAB menyediakan operator dan karakter khusus yang sering dipakai:

Operator	Kegunaan
+	Operator penambah
-	Operator pengurang
*****	Operator pengali scalar atau matriks
.*	Operator pengali array
^	Operator pangkat scalar dan matriks
.^	Operator pangkat array
****	Operator pembagi kiri
/	Operator pembagi kanan
.	Operator pembagi kiri array
./	Operator pembagi kanan array
:	Penghasil elemen secara berurut dan menampilkan pada isi sebuah kolom atau lajur
( )	Penunjuk lampiran argument fungsi dan array
[ ]	Melampirkan elemen-elemen
.	Titik decimal
…	Operator garis penerus
,	Pemisah atatemen dan elemen_elemen dalam lajur
;	Pemisah atatemen dan elemen_elemen dalam kolom dan menyembunyikan keluaran hasil fungsi
%	Menunjukkan sebuah komentar dan menentukan format



1.3.2 Variabel dan Konstanta Khusus

MATLAB menyediakan operator dan karakter khusus yang sering dipakai:

Nama	Arti
Ans	Hasil
Eps	Ketepatan titik ketelitian pecahan
I,j	Satuan imajinir $\sqrt{-1}$
Inf	Tak berhingga
NaN	Hasil numerik yang tidak terdefenisi
Pi	Bilangan π

1.4 Variabel-variabel Matlab

Pada Jendela perintah window MATLAB, setiap variable adalah sebuah array atau matriks.

Contoh :

```
>> x = 3                          % menentukan dan mengisi x dengan sebuah nilai
```

Bila tombol eksekusi ditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>x =3
```

MATLAB menciptakan sebuah matriks 1 x 1 yang dinamai x dan menyimpan nilai 3 sebagai elemennya.

Contoh lain :

```
>>x = sqrt(16)   % menentukan dan mengisi x dengan sebuah fungsi
```



Bila tombol eksekusiditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>x =4
```

Catat bahwa : sekali sebuah variable telah dimasukan dalam system, maka kita dapat mengambilnya nanti. Variabel harus mempunyai nilai sebelum mereka digunakan.

Bila sebuah ekspresi menghasilkan sebuah hasil yang bukan milik sebuah variable, system menyimpan hasilnya dalam sebuah variable yang diberi namaans, yang mana dapat digunakan.

Contoh :

```
>>sqrt(78)
```

Bila tombol eksekusiditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>ans =8.8318
```

Anda dapat menggunakan variable ans ini :

```
>>9876/ans
```

Bila tombol eksekusiditekan, MATLAB mengeksekusi itu secara langsung dan menghasilkan :

```
>>ans =1.1182e+03
```

Contoh:

```
>>x = 7 * 8;y = x * 7.89
```

Bila tombol eksekusiditekan, MATLAB mengeksekusi dan menghasilkan :

```
>>y =441.8400
```



1.4.1 Penulisan Statemen Banyak

Anda dapat meletakkan banyak statemen dalam baris yang sama :

```
>>a = 2; b = 7; c = a * b
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>c =14
```

Jika Anda lupa nama variable, dapat gunakan :

```
>>who
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan variable Anda adalah :

```
>>a ans b c x y
```

Perintah clear menghapus semua variable (atau tertentu) dari memori.

```
>>clear x        % akan menghapus variable x
>>clear          % akan menghapus semua variable
```

1.4.2 Penulisan Statemen Panjang

Penempatan statemen yang panjang dapat diperluas ke baris berikutnya dengan menggunakan tanda ellipse (...), contoh :

```
>>initial_velocity = 0;
>>acceleration = 9.8;
>>time = 20;
>>final_velocity = initial_velocity ...
>>+ acceleration * time
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>final_velocity =196
```



1.4.3 Perintah Format

MATLAB defaultnya menampilkan bilangan dengan empat tempat decimal dibelakang koma, ini disebut format short. Walaupun demikian , jika Anda ingin lebih presisi, Anda perlu menggunakan perintah format long.

```
>>format long
>>x = 7 + 10/3 + 5 ^ 1.2
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>x =17.231981640639408
```

Contoh lain :

```
>>format short
>>x = 7 + 10/3 + 5 ^ 1.2
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>x =17.2320
```

Perintah formatbank mendekatkan bilangan menjadi 2 decimal dibelakang koma.

```
>>format bank
>>daily_wage = 177.45;
>>weekly_wage = daily_wage * 6
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>weekly_wage =1064.70
```

1.4.4 Membuat Vektor

Sebuah vector adalah sebuah array dimensi satu.. MATLAB mengijinkan membuat dua jenis vector yaitu Vektor baris Vektor kolom.

Vektor baris dibuat dengan meletakkan himpunan elemen-elemen dalam kurung kotak, menggunakan spasi atau koma untuk membatasi elemen- elemen.

```
>>r = [7 8 9 10 11]
```



MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>r =Columns 1 through 4
7 8 9 10
Column 5
11
```

Contoh lain :

```
>>r = [7 8 9 10 11];
>>t = [2, 3, 4, 5, 6];
>>res = r + t
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>res =Columns 1 through 4
        9 11 13 15
        Column 5
        17
```

Vektor kolom dibuat dengan meletakkan himpunan elemen-elemen dalam kurung kotak, menggunakan titik koma untuk membatasi elemen- elemen.

```
>>c = [7; 8; 9; 10; 11]
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>c = 7
8
9
10
11
```

1.4.5 Membuat Matriks

Sebuah matriks adalah sebuah array bilangan dua dimensi. Dalam MATLAB, sebuah matriks dibuat dengan meletakkan setiap baris sebagai sebuah sekuen bilangan yang dipisahkan oleh sapasi atau koma dan akhir dari baris diakhiri oleh titik koma.Contoh matriks 3 x 3 :

```
>>m = [1 2 3; 4 5 6; 7 8 9]
```

MATLAB akan mengeksekusi statemen diatas dan menghasilkan :

```
>>m =1 2 3
      4 5 6
      7 8 9
```

1.5 Perintah-perintah Matlab

MATLAB adalah sebuah program interaktif untuk komputasi numeric dan visualisasi data. Anda dapat memasukan sebuah perintah dengan mengetiknya pada prompt MATLAB “>>" pada jendela Perintah. Dalam sesi ini, tersedia daftar perintah yang biasa dipakai.

1.5.1 Perintah untuk mengatur sebuah sesi

MATLAB menyediakan bermacam-macam perintah untuk mengatur sebuah sesi.

Perintah	Maksud
Clc	Membersihkan layar jendela perintah
Clear	Menghapus variable dari memori
Exist	Memeriksa keberadaan file atau variable
Global	Menentukan variable menjadi global
Help	Mencari sebua topic pertolongan
Quit	Menghentikan MATLAB.
Who	Menampilkan daftar variable yang sedang aktif.
Whos	Menampilkan daftar variable yang sedang aktif (tampilan panjang)



1.5.2 Perintah-perintah Input Output

MATLAB menyediakan perintah-perintah yang berhubungan dengan input output.

Perintah	Maksud
disp	Menampilkan isi dari sebuah arra atau string
fscanf	Membaca data terformat dari file
Format	Mengontrol format tampilan layar
Fprintf	Menulis data terformat ke layar atau file.
Input	Menampilkan prompt dan menunggu input
;	Membaca data terformat dari file

Tabel berikut menunjukkan perintah-perintah yang digunakan untuk format bilangan dan string.

Kode Format	Maksud
%s	Format sebagai string.
%d	Format sebagai bilangan bulat.
%f	Format sebagaibilangan pecahan.
%e	Format sebagaibilangan pecahandalam notasi scientific.
%g	Format in the most compact form: %f or %e.
\n	Disisipi pada baris baru dalam format string.
\t	Disisipi pada tab baru dalam format string.



Tabel berikut menunjukkan perintah-perintah yang digunakan untuk format.

Fungsi Format	Tampilan
format short	4 digit desimal (default).
format long	16 digit desimal.
format short e	5digit desimal plus exponen.
format long e	16 digits plus exponents.
format bank	2 digit desimal.
format +	Positip, negatip, ataunol.
format rat	Pendekatan Rational.
format compact	Merapatkan beberapa baris.



1.5.3 Perintah Vektor, Matriksdan Array

Tabel berikut menunjukkan perintah-perintah yang digunakan untuk bekerja dengan vector, matriks dan array.

Perintah	Maksud
length	Menghitung jumlah elemen.
linspace	Membuat jarak vector secara linier.
logspace	Membuat jarak vector secara logarimis.
max	Menghasilkan elemen yang terbesar.
min	Menghasilkan elemen yang terkecil.
prod	Perkalian vektor setiap kolom.
reshape	Mengubah ukuran.
size	Menghitung ukuran array.
sort	Mensortir setiap kolom.
sum	Menjumlah setiap kolom.
eye	Membuat sebuah matriks identitas.
ones	Membuat sebuah matriks satu.
zeros	Membuat sebuah matriks nol.
cross	Menghitung matriks perkalian silang.
dot	Menghitung perkalian matriks.
Det	Menghitung determinan sebuah matriks.
inv	Menghitung inversi sebuah matriks.
rank	Menghitung rank sebuah matriks.
cell	Creates cell array.
celldisp	Menampilkan sel array.
num2cell	Mengkonversi bilangan array ke sel array.



1.5.4 Perintah Menggambar

Tabel berikut menunjukkan perintah-perintah yang digunakan untuk bekerja gambar.

Perintah	Maksud
axis	Menentukan batas aksis.
grid	Menampilkan garis bantu.
plot	Menghasilkan gambar xy.
print	Mencetak gambar atau menyimpan ke file.
title	Memberikan teks pada judul gambar
xlabel	Menulis teks pada aksis x.
ylabel	Menulis teks pada aksis y.
axes	Membuat objek aksis.
close	Menutup gambar yang aktif.
close all	Menutup semua gambar.
figure	Membuat gambar yang baru.
gtext	Membuat label yang pada posisi mouse.
hold	Menampilkan gambar baru padagambar yang aktif.
legend	Menampilkan nama objek gambar pada posisi mouse.
refresh	Menampilkan ulang objek pada jendela gambar.
subplot	Membuat gambar pada jendela-jendela kecil (subwindows).
text	Membuat teks pada gambar.
bar	Membuat gambar chart balok.
loglog	Membuat gambar log-log.
polar	Membuat gambar polar.
semilogx	Membuat gambar semilog. (logarithmic abscissa).
semilogy	Membuat gambar semilog. (logarithmic ordinate).
stairs	Membuat gambar tangga.
stem	Membuat gambar jarum



Rangkuman

- MATLAB membantu memecahkan berbagai masalah matematis yang kerap kita temui dalam bidang teknis secara cepat.
- Struktur awal aplikasi MATLAB terdiri atas jendela workspace, command history dan comand editor.
- Matlab memiliki aturan penulisan script yang di terapkan dalam penulisan perintah, variabel, karakter dan konstanta.
- Fungsi perintah dalam MATLAB terdiri atas empat kelompok yaitu perintah mengatur sebuah sesi, perintah input-output, perintah menggambar dan perintah untuk membuat vektor, matriks maupun array.



Tugas

1. Pahami setiap perintah dan Lakukan praktek pada komputer, semua tutorial diatas.
2. Buatlah latihan sendiri dengan mengubah-ubah tutorial diatas

Tes Formatif



KEGIATAN 2

1.6 Berkas.m (M Files)

MATLAB mengijinkan penulisan dua macam file program :

- Scripts –berkas script adalah file program dengan ekstensi .m. Dalam file ini Anda menulis perintah-perintah secara seri, dimana perintah-perintah ini akan dieksekusi secara bersama-sama. Fungsi tidak dapat menerima input dan tidak menghasilkan hasil.
- Functions –berkas fungsi adalah juga file program dengan ekstensi .m. Fungsi dapat menerima input dan menghasilkan hasil.

Anda dapat menggunakan editor MATLAB atau editor teks untuk membuat file .m. Sebuah file script terdiri dari banyak baris sekuensial ari perintah-perintah dan fungsi yang dipanggil. Sebuah file script dapat dijalankan dengan mengetikkan namanya pada baris jendela perintah.

1.6.1 Mengunakan Prompt Perintah

Jika menggunakan prompt perintah, ketik edit pada prompt perintah. MATLAB akan membuka editor teks, dan dapat langsung mengetik perintah-perintah yang diinginkan dan kemudian menulis nama file (ekstensi .m).

```
>>edit
```

atau

```
>>edit <filename>
```

Perintah diatas akan membuat sebuah file kerja yang berada pada direktori default. Jika ingin menyimpan semua file-file program dalam sebuah folder tertentu, tersedia juga untuk path keseluruhan.

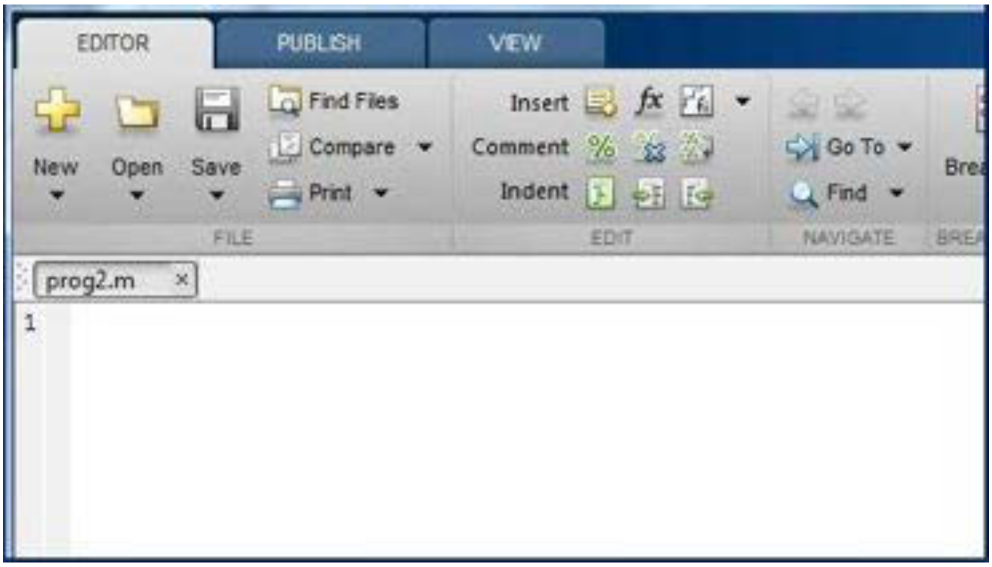
Buat sebuah folder yang diberi nama progs. Ketik perintah berikut pada prompt perintah (>>) :

```
>>mkdir progs          % membuat direktori progs padadirektoridefault
>>chdir progs          % merubah direktori default menjadi progs
>>edit  prog1.m % membuat sebuah file m bernama prog1.m
```

Jika membuat file untuk pertama kali,prompts MATLAB akan mengkonfirmasi, ketik Yes.

1.6.2 Membuat dan Menjalankan File ScriptMengunakan IDE

Jika menggunakan IDE, pilih NEW -> Script. Ini juga akan membuka editor dan membuat sebuah file dengan nama Untitled. Anda dapat menyimpan dengan nama baru setelah penuliskan perintah-perintahnya.



Gambar 1.3Jendela IDE MATLAB.

Ketikan perintah-perintah berikut ini pada editor:

```
NoOfStudents = 6000;

TeachingStaff = 150;

NonTeachingStaff = 20;

Total = NoOfStudents + TeachingStaff + NonTeachingStaff;

disp(Total);
```

Setelah membuat dan menyimpan file ini, Anda dapat menjalankannya dalam dua cara :

Tekan tombol Run pada jendela editor atau hanya mengetik nama file (tanpa ekstensi) pada prompt perintah

```
>>prog1
```



Jendela perintah akan menampilkan hasil :

```
>>6170
```

Contoh lain :

Buat file script dan ketikan perintah-perintah dibawah ini :

```
a = 5; b = 7;
c = a + b
d = c + sin(b)
e = 5 * d
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>c =
    12
>>d =
   12.6570
>>e =
   63.2849
```

1.7 Vektor

Vektor adalah sebuah array dimensi satu dari bilangan.MATLAB mengijinkan membuat dua tipe vector

1.7.1 Vektor Baris:

Vektor baris dibuat oleh himpunan tertutup dari elemen-elemen dalam kurung kotak, menggunakan spasi atau koma untuk memisahkan elemen-elemen.

```
r = [7 8 9 10 11]
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>r =

  Columns 1 through 5

     7        8        9        10        11
```

1.7.2 Vektor Kolom:

Vektor baris dibuat oleh himpunan tertutup dari elemen-elemen dalam kurung kotak, menggunakan titik-koma untuk memisahkan elemen-elemen.

```
c = [7;  8;  9;  10]
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>c =

       7

       8

       9

      10
```

1.7.3 Referensi Element dari sebuah Vektor

Anda dapat mereferensi satu atau lebih elemen-elemen dari sebuah vector.Komponen ke I dari vector v direferensikan sebagai v(i). Contoh :

```
v = [ 1; 2; 3; 4; 5; 6];

          v(3)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

     3
```

Bila Anda mereferensi sebuah vector dengan sebuah titik dua, semua komponen dari vector akan ditampilkan.

```
v = [ 1; 2; 3; 4; 5; 6];    % creating a column vector of 6 elements

v(:)
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

    1

    2

    3

    4

    5

    6
```

1.8 Matriks

Matriks adalah sebuah array dua dimensi dari bilangan. Dalam MATLAB Anda membuat sebuah matriks dengan memasukkan elemen-elemen dalam setiap baris dengan koma atau spasi sebagai pemisah dan menggunakan titik koma sebagai tanda akhir dari setiap baris.

Contoh membuat matriks a dimensi 4 x 5:

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8]
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a =

    1    2    3    4    5

    2    3    4    5    6

    3    4    5    6    7

    4    5    6    7    8
```

1.8.1 Referensi Element-elemen Matriks

Untuk mereferensi sebuah elemen dalam baris m dan kolom n dari sebuah matriks mx, tuliskan :

```
>>mx(m, n);
```



Contoh untuk mereferensi elemen baris ke dua dan kolom ke lima, ketikkan :

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
a(2,5)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =
   6
```

Untuk mereferensi semua elemen dalam kolom ke m, ketikkan A(:,m) :

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
v = a(:,4)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>v =
   4
   5
   6
   7
```

Contoh membuat matriks kecil yang elemen-elemennya diambil dari kolom kedua dan ketiga matriks besar.

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];
a(:, 2:3)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =
   2   3
   3   4
   4   5
   5   6
```



Cara yang sama untuk membuat sub matriks dari sub matriks.

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];

a(:, 2:3)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

   2    3

   3    4

   4    5

   5    6
```

1.8.2 Menghapus sebuah Baris atau Kolom dalam

Anda dapat menghapus seluruh isi sebuah baris atau kolom sebuah matriks dengan menuliskan [] pada baris atau kolom yang diinginkan.

Contoh :

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];

a( 4 , : ) = []
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a =

   1    2    3    4    5

   2    3    4    5    6

   3    4    5    6    7
```

Selanjutnya :

```
a = [ 1 2 3 4 5; 2 3 4 5 6; 3 4 5 6 7; 4 5 6 7 8];

a(: , 5)=[]
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a =

    1    2    3    4

    2    3    4    5

    3    4    5    6

    4    5    6    7
```

1.9 Array

Dalam MATLAB, semua variable tipe data adalah array multi dimensi. Vector adalah sebuah array dimensi satu dan matriks adalah array dimensi dua atau lebih.

1.9.1 Arrays Khusus dalam MATLAB

Fungsi zeros() membuat sebuah array semua nol :

Contoh :

```
zeros(5)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

    0    0    0    0    0

    0    0    0    0    0

    0    0    0    0    0

    0    0    0    0    0

    0    0    0    0    0
```



Fungsi ones() membuat sebuah array semua satu :

Contoh :

```
ones(4,3)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
ans =

   1   1   1

   1   1   1

   1   1   1

   1   1   1
```

Fungsi eye() membuat sebuah array identitas :

Contoh :

```
eye(4)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

   1   0   0   0

   0   1   0   0

   0   0   1   0

   0   0   0   1
```



1.9.2 Arrays Multi Dimensi

Sebuah array mempunyai lebih dari dua dimensi disebut array multi dimensi.Array multi dimensi dalam MATLAB adalah sebuah pengembangan dari matriks dimensi normal.

Contoh :

```
a = [7 9 5; 6 1 9; 4 3 2]
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a =
   7   9   5
   6   1   9
   4   3   2
```

Array adalah sebuah array dimensi tiga, dapat ditambahkan sebuah dimensi ketiga pada a dengan cara seperti ini :

```
a(:, :, 2)= [ 1 2 3; 4 5 6; 7 8 9]
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a(:,:,1) =
   7   9   5
   6   1   9
   4   3   2
>>a(:,:,2) =
   1   2   3
   4   5   6
   7   8   9
```

Kita dapat membuat array multi dimensi menggunakan fungsi-fungsi ones(), zeros() atau rand().

Contoh :

```
b = rand(4,3,2)
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>b(:,:,1) =
    0.0344    0.7952    0.6463
    0.4387    0.1869    0.7094
    0.3816    0.4898    0.7547
    0.7655    0.4456    0.2760
>>b(:,:,2) =
    0.6797    0.4984    0.2238
    0.6551    0.9597    0.7513
    0.1626    0.3404    0.2551
    0.1190    0.5853    0.5060
```

Kita dapat menggunakan fungsicat() untuk membuat array multi dimensi.

Sintaks fungsicat():

```
B = cat(dim, A1, A2...)
```

dimana,B adalah array batu yang akan dibuat, A1, A2, ... adalah array yang akan di gabungkan dan dim dalah dimensi array hasil penggabungan

```
a = [9 8 7; 6 5 4; 3 2 1];
b = [1 2 3; 4 5 6; 7 8 9];
c = cat(3, a, b, [ 2 3 1; 4 7 8; 3 9 0])
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>c(:,:,1) =

    9    8    7

    6    5    4

    3    2    1

>>c(:,:,2) =

    1    2    3

    4    5    6

    7    8    9

>>c(:,:,3) =

    2    3    1

    4    7    8

    3    9    0
```



Rangkuman

- MATLAB memiliki dua macam penulisan file program yaitu script dan function.
- Script dan Fuctions merupakan file program dengan ekstensi .m. dimanadituliskan perintah-perintah secara terstruktur dan berurutan.
- Vektor adalah sebuah array dimensi satu dari suatu bilangan. Vektor terbagi atas vektor array baris dan vektor array kolom.
- Matrik adalah array dua dimensi dari sekumpulan bilangan. Contoh matriks 2x2 (dua baris dan dua kolom), 2x3 (dua baris dan tiga kolom), 4x4 (empat baris dan empat kolom) dll.
- Array multi dimensi merupakan array yang memiliki dimensi dua atau lebih.
- Beberapa fungsi khusus dari array antara lain fungsi “zeros” (array beranggotakan nilai 0), “ones” (array beranggotakan nilai 1) dan “eye” (array beranggotakan nilai 1 secara diagonal-disebut juga matrik identitas karena nilai berapapun jika dikalikan dengan matrik identitas akan menghasilkan nilai awal/sendiri dari matrik tersebut).



Tugas

1. Pahami setiap perintah dan Lakukan praktek pada komputer, semua tutorial diatas.
2. Buatlah latihan sendiri dengan mengubah-ubah tutorial diatas

Tes Formatif



KEGIATAN 3

1.10 Tipe Data dalam MATLAB

MATLAB menyediakan 15 tipe data fundamental.Setiap tipe data menyimpan data tersebut dalam format sebuah matriks atau array.Ukuran matriks atau array adalah minimum 0 x 0 dan ini dapat berkembang.

Tabel berikut menunjukkan tipe-tipe data dalam MATLAB :

Tipe Data	Penjelasan
int8	8-bit bertanda integer
uint8	8-bit tidak bertanda integer
int16	16-bit bertandainteger
uint16	16-bit tidak bertandainteger
int32	32-bit bertandainteger
uint32	32-bit tidak bertandainteger
int64	64-bit bertandainteger
uint64	64-bit tidak bertandainteger
Single	Data numeric presisi tunggal
Double	Data numeric presisi ganda
Logical	Nilai logika 1 atau 0, hasil masing-masing benaratau salah
char	Data karakter (strings disimpansebagai vektor karakter)
cell array	Seldari indeks array
function handle	Pointer mengarah pada sebuah fungsi



Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
str = 'Hello World!'
n = 2345
d = double(n)
un = uint32(789.50)
rn = 5678.92347
c = int32(rn)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>str =
        Hello World!
>>n =
        2345
>>d =
        2345
>>un =
        790
>>rn =
        5.6789e+03
>>c =
        5679
```



1.10.1 Konversi Tipe Data

MATLAB menyediakan bermacam-macam fungsi untuk mengkonversi dari sebuah tipe data menjadi yang lain. Tabel berikut ini menunjukkan fungsi konversi tipe data:

Fungsi	Maksud
char	Mengkonversike array karakter (string)
int2str	Mengkonversi data integer ke string
mat2str	Mengkonversi matriks ke string
num2str	Mengkonversibilangan ke string
str2double	Mengkonversi string ke double nilai presisi
str2num	Mengkonversi string ke bilangan
bin2dec	Mengkonversi string bilangan binary ke bilangan decimal
dec2bin	Mengkonversi desimal kestring bilangan binary
dec2hex	Mengkonversi desimal kestring bilanganhexadecimal
hex2dec	Mengkonversistring bilanganhexadesimal ke decimal
cell2mat	Mengkonversiarray selkearray numeric
cellstr	Membuat array sel stringdari array karakter
mat2cell	Mengkonversi array ke array sel



1.10.2 Penentuan Tipe Data

MATLAB menyediakan bermacam-macam fungsi untuk identifikasi tipe data sebuah variable.

Tabel berikut menyediakan fungsi-fungsi tipe data sebuah variable :

Fungsi	Maksud
Is	Mendeteksi keadaan
Isa	Determine jika input adalaha object klas tertentu
iscell	Menentukanapakah input adalah array sel
iscellstr	Menentukanapakah input adalah array selstring
ischar	Menentukanapakah item adalah array karakter
isfield	Menentukanapakah input adalahbidang array structure
isfloat	Menentukanjika input adalah array titik pecahan
isinteger	Menentukanjika input adalaharray integer
islogical	Menentukanjika input adalaharray logical
isnumeric	Menentukanjika input adalaharray numeric
isobject	Menentukanjika input adalahobject MATLAB
isreal	Memeriksa jika input adalaharray real
isscalar	Menentukanapakah input adalahscalar
isstr	Menentukanapakah input adalaharray character
isstruct	Menentukanapakah input adalaharray structure
isvector	Menentukanapakah input adalah vector



Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
x = 3
isinteger(x)
isfloat(x)
isvector(x)
isscalar(x)
isnumeric(x)
x = 23.54
isinteger(x)
isfloat(x)
isvector(x)
isscalar(x)
isnumeric(x)
x = [1 2 3]
isinteger(x)
isfloat(x)
isvector(x)
isscalar(x)
x = 'Hello'
isinteger(x)
isfloat(x)
isvector(x)
isscalar(x)
isnumeric(x)
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>x =
    3
>>ans =
    0
>>ans =
    1
>>ans =
    1
>>ans =
    1
>>ans =
    1
>>x =
  23.5400
>>ans =
    0
>>ans =
    1
>>ans =
    1
>>ans =
    1
>>ans =
    1
>>x =
    1    2    3
>>ans =
    0
```



```
>>ans =

    1

>>ans =

    1

>>ans =

    0

>>x =

Hello

>>ans =

    0

>>ans =

    0

>>ans =

    1

>>ans =

    0

>>ans =

    0
```

1.11 Operator MATLAB

Sebuah operator adalah sebuah symbol yang memberitahu compiler untuk melakukan manipulasi matematika atau logika tertentu.MATLAB didisain untuk melakukan operasi matriks atau array. Oleh sebab itu, operatorsdalam MATLAB bekerja untuk data scalar dan non scalar.

1.11.1 Operator Arithmetik

MATLAB mengijinkan dua tipe berbeda dari operasi aritmatika :

- Operasi Matriks arithmetika
- Operasi Array arithmetika



Operasi matriks arithmetikasama seperti operasi pada aljabar linier. Operasi Array dieksekusi elemen perelemen, baik pada dimensi satu maumun pada dimensi banyak.

Operator matriksdan array dibedakan oleh tanda titik (.). Untuk operasi penjumlahan dan pengurangan adalah berlaku sama untuk matriks dan array.

Operator	Kegunaan
+	Operator penambah
-	Operator pengurang
*	Operator pengali scalar atau matriks
.*	Operator pengali array
^	Operator pangkat scalar dan matriks
.^	Operator pangkat array
\	Operator pembagi kiri
/	Operator pembagi kanan
.\	Operator pembagi kiri array
./	Operator pembagi kanan array

Contoh :

```
a = 10;
b = 20;
c = a + b
d = a - b
e = a * b
f = a / b
g = a \ b
x = 7;
y = 3;
z = x ^ y
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>c =
        30
>>d =
        -10
>>e =
        200
>>f =
        0.5000
>>g =
        2
>>z =
        343
```

1.11.2 Operator Relasi

Operator relasi dapat juga bekerja pada kedua data scalar dan non scalar. Operator relasi untuk array elemen per elemen dibandingkan antara dua array dan menghasilkan sebuah array dengan ukuran yang sama berisi elemen-elemen yang diset logika 1 bila benar dan logika 0 bila salah.

Tabel berikut menyediakan operator relasi :

Operator	Penjelasan
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
~=	Not equal to



Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 100;
b = 200;
if (a >= b)
max = a
else
max = b
end
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>max =
  200
```

1.11.3 Operator Logika

MATLAB menawarkan dua tipe operator logika:

Operator logical Element mengoperasikan elemen per elemen pada array logika.Simbol-simbol&, |, dan ~ adalah operator array logika AND, OR, dan NOT.

Operatorslogical rangkaian singkat pengijinkanrangkaian singkat pada operator logika. Simbol-simbol&& and || adalah operators logical rangkaian singkat AND and OR.

Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 5;
b = 20;
if ( a && b )
disp('Line 1 - Condition is true');
end
if ( a || b )
disp('Line 2 - Condition is true');
```



```
end
   % lets change the value of  a and b
   a = 0;
   b = 10;
if ( a && b )
disp('Line 3 - Condition is true');
else
disp('Line 3 - Condition is not true');
end
if (~(a && b))


disp('Line 4 - Condition is true');
end
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>Line 1 - Condition is true
>>Line 2 - Condition is true
>>Line 3 - Condition is not true
>>Line 4 - Condition is true
```

1.11.4 Operasi Bitwise

Operator Bitwise bekerja pada bit-bitnya dan membentuk oleh operasi bit.

Tabel kebenaran untuk &, |, and ^ :

P	q	p & q	p l q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1



Asumsikan jika A = 60; dan B = 13;

A = 0011 1100

B = 0000 1101

A&B = 0000 1100

A|B = 0011 1101

A^B = 0011 0001

~A = 1100 0011

Tabel berikut ini menunjukkan operasi bitwise:

Fungsi	Maksud
bitand(a, b)	Bit-wise AND integer a dan b
bitor(a, b)	Bit-wise OR integer a dan b
bitshift(a, k)	Menggeser kekiri k bit, equivalent dengan pengalian 2k. Menggeser kekanan atau dibagi oleh 2|k|.
bitxor(a, b)	Bit-wise XOR integer a dan b

Contoh

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 60; % 60 = 0011 1100

b = 13; % 13 = 0000 1101

c = bitand(a, b)      % 12 = 0000 1100

c = bitor(a, b)      % 61 = 0011 1101

c = bitxor(a, b)      % 49 = 0011 0001

c = bitshift(a, 2)    % 240 = 1111 0000 */

c = bitshift(a,-2)    % 15 = 0000 1111 */
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>c =

    12

>>c =

    61

>>c =

    49

>>c =

   240

>>c =

    15
```

1.11.5 Operasi Set

MATLAB menyediakan macam-macam fungsi untuk operasi set, seperti gabungan, irisan dan pengujian anggota himpunan

Tabel berikut ini menunjukkan operasi set :

Fungsi	Penjelasan
intersect(A,B)	Set irisan dua array A dan B.
setdiff(A,B)	Set perbedaan dua array A dan B.
Setxor	Set exclusive OR dua array A dan B
Union	Set union of two arrays



Contoh

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = [7 23 14 15 9 12 8 24 35]

b = [ 2 5 7 8 14 16 25 35 27]

u = union(a, b)

i = intersect(a, b)

s = setdiff(a, b)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>> a =

    7   23   14   15    9   12    8   24   35

>> b =

    2    5    7    8   14   16   25   35   27

>> u =

  Columns 1 through 11

    2    5    7    8    9   12   14   15   16   23   24

  Columns 12 through 14

   25   27   35

>> i =

    7    8   14   35

>> s =

    9   12   15   23   24
```



Rangkuman

- Tipe data pada MATLAB terkelompok menjadi 15 fundamen. Secara garis besar adalah tipe data integer 8 bit, 16 bit, 32 bit dan 64 bit, numerik single (pecahan), numerik double, char (teks/karakter), array dan logika.
- Hal yang harus diperhatikan dalam melakukan operasi matematika adalah tipe data yang digunakan. Untuk itu diperlukan fungsi untuk mengkonversi dari sebuah tipe data menjadi tipe data lain. Sebagai contoh kita hendak mengalikan nilai biner “0111” dengan nilai desimal “10”. Maka diperlukan konversi tipe data biner ke desimal dengan menggunakan perintah “bin2dec”.
- MATLAB memiliki beberapa operator yaitu :
    - Operator aritmatika
    - Operator relasi
    - Operator Logika
    - Operator bitwise
    - Operator Set



Tugas

1. Pahami setiap perintah dan Lakukan praktek pada komputer, semua tutorial diatas.
2. Buatlah latihan sendiri dengan mengubah-ubah tutorial diatas

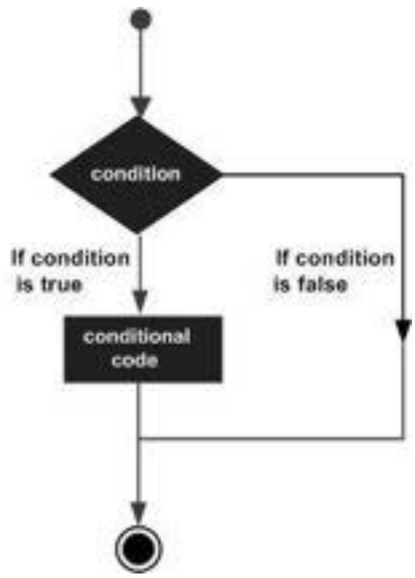
Tes Formatif



KEGIATAN 4

1.12 Pembuatan Keputusan MATLAB

Struktur pembuatan keputusan mengharuskan bahwa pemogram menentukan satu atau lebih kondisi yang akan dievaluasi atau diuji oleh program, sesuai dengan pernyataan yang akan dieksekusi jika kondisi ditentukan benar dan pernyataan lain kan dieksekusi bila kondisi salah.Berikut ini adalah bentuk umum sebuah struktur pembuatan keputusan :



Gambar 1. 4Diagram Alir if…end.

MATLAB menyediakan tipe pembuatan keputusan berikut ini :

Statemen	Penjelasan
if ... end statement	If ... end statement,terdiri dari sebuah ekspresi boolean diikuti oleh satu atau lebihpernyataan.
if...else...end statement	If statement dapat diikuti oleh sebuah opsional else statement, dimana dieksekusi bila ekspresi boolean salah.
If... elseif...elseif...else...end statements	If statement dapat diikuti oleh sebuah opsional else statement elseif... dan sebuah else statement, berguna untuk menguji bermacam-macam kondisi.
nested if statements	If … elseif statement di dalam if … elseif statement.



Pembuatan Keputusan if ... end

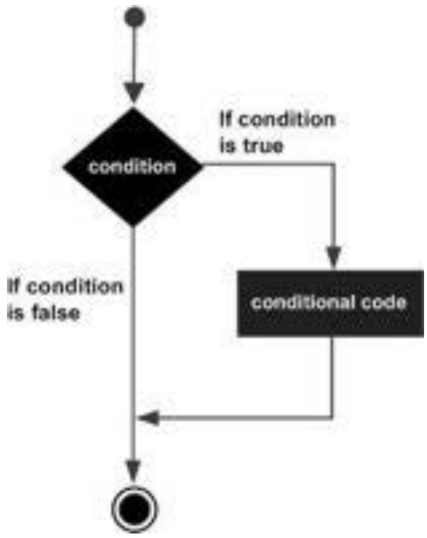
Sintaks :

```
if<expression>

% statement akan dieksekusi bila ekspresi boolean benar

<statements>

end
```



Gambar 1. 5Diagram Alir if…end.

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 10;

% check the condition using if statement

if a < 20

   % if condition is true then print the following

fprintf('a is less than 20\n' );

end

fprintf('value of a is : %d\n', a);
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a is less than 20

>>value of a is : 10
```

1.12.1 Pembuatan Keputusan if ... else end

Sintaks :

```
if<expression>

        % statement(s) akan dieksekusi bila ekspresi boolean benar

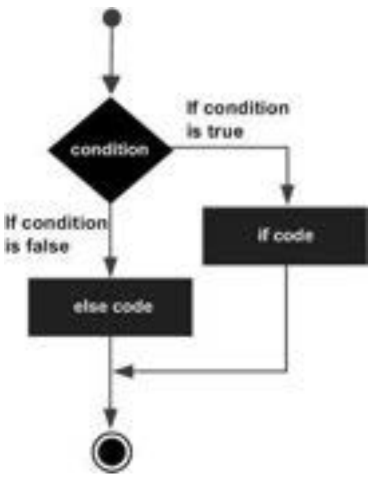
<statement(s)>

else

<statement(s)>

        % statement(s) akan dieksekusi bila ekspresi boolean salah

end
```



Gambar 1. 6Diagram Alir if…end.

Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 100;

% check the boolean condition

if a < 20
```



```
        % if condition is true then print the following
fprintf('a is less than 20\n' );
else
       % if condition is false then print the following
fprintf('a is not less than 20\n' );
end
fprintf('value of a is : %d\n', a);
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>a is not less than 20
>>value of a is : 100
```

1.12.2 Pembuatan Keputusan if ... elseif … else end

Sintaks :

```
if<expression 1>
          % akan dieksekusi bila ekspresi Boolean 1 benar
<statement(s)>
elseif<expression 2>
          % akan dieksekusi bila ekspresi Boolean 2 benar
<statement(s)>
Elseif <expression 3>
          % akan dieksekusi bila ekspresi Boolean 3 benar
<statement(s)>
else
          % akan dieksekusi bila semua ekspresi salah
<statement(s)>
End
```



Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 100;
%check the boolean condition
if a == 10
     % if condition is true then print the following
fprintf('Value of a is 10\n' );
elseif( a == 20 )
    % if else if condition is true
fprintf('Value of a is 20\n' );
elseif a == 30
     % if else if condition is true
fprintf('Value of a is 30\n' );
else
     % if none of the conditions is true '
fprintf('None of the values are matching\n');
fprintf('Exact value of a is: %d\n', a );
end
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>> None of the values are matching
>>Exact value of a is: 100
```



1.12.3 Pembuatan Keputusan If … else end Bercabang

Sintaks :

```
if<expression 1>
        % akan dieksekusi bila ekspresi boolean1 benar
if<expression 2>
        % akan dieksekusi bila ekspresi Boolean 2 benar
end
end
```

Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 100;
b = 200;
    % check the boolean condition
if( a == 100 )
      % if condition is true then check the following
if( b == 200 )
          % if condition is true then print the following
fprintf('Value of a is 100 and b is 200\n' );
end
end
fprintf('Exact value of a is : %d\n', a );
fprintf('Exact value of b is : %d\n', b );
```

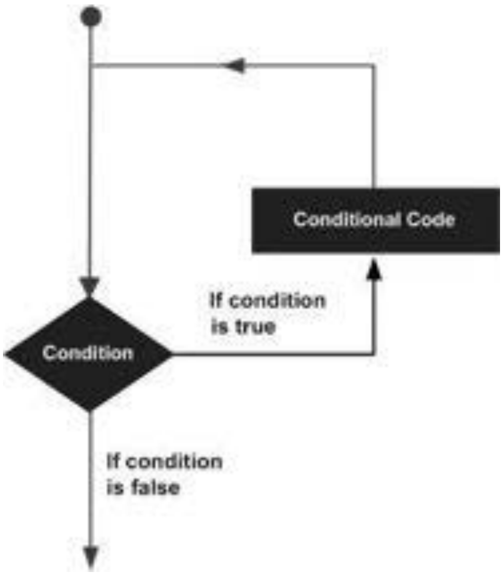
Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>Value of a is 100 and b is 200
>>Exact value of a is : 100
>>Exact value of b is : 200
```



1.13 Tipe Pengulangan MATLAB

Sebuah pernyataan pengulangan mengijinkan untuk mengeksekusi sebuah pernyataan atau kelompok pernyataan berkali-kali dan mengikuti format umum pernyataan pengulangan berikut ini :



Gambar 1. 7Diagram Alir if…end.

Tipe Pengulangan	Penjelasan
while loop	Pengulangan sebuah atau sekelompok pernyataan sementara kondisi benar. Itu menguji sebelum mengeksekusi pernyataan.
for loop	Mengeksekusi sebuah sekuensial pernyataan berkali-kali.
nested loops	Pengulangan di dalam pengulangan.



1.13.1 Pengulangan while … end

Sintaks :

```
while<expression>
        <statements>
end
```

Pengulangan akan dilakukan terus menerus sepanjang ekspresi benar..

Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
a = 10;
        % while loop execution
while( a < 20 )
fprintf('value of a: %d\n', a);
  a = a + 1;
end
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>value of a: 11
>>value of a: 12
>>value of a: 13
>>value of a: 14
>>value of a: 15
>>value of a: 16
>>value of a: 17
>>value of a: 18
>>value of a: 19
```



1.13.2 Pengulangan for…end

Sintaks :

```
for index = values
        <program statements>
end
```

Pengulangan akan dilakukan terus menerus sebanyak values kali.

Contoh :

Buatlah sebuah file script dengan kode seperti dibawah ini :

```
for a = 10:20
fprintf('value of a: %d\n', a);
end
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>value of a: 10
>>value of a: 11
>>value of a: 12
>>value of a: 13
>>value of a: 14
>>value of a: 15
>>value of a: 16
>>value of a: 17
>>value of a: 18
>>value of a: 19
>>value of a: 20
```



1.13.3 Pengulangan for…end Bercabang

Sintaks :

```
for m = 1:j
for n = 1:k
<statements>;
end
end
```

Pengulangan didalam pengulangan akan dilakukan terus menerus sebanyak m dan n kali.Buatlah sebuah file script dengan kode seperti dibawah ini :

```
for i=2:100
for j=2:100
if(~mod(i,j))
break; % if factor found, not prime
end
end
if(j > (i/j))
fprintf('%d is prime\n', i);
end
end
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>2 is prime
>>3 is prime
>>5 is prime
>>7 is prime
>>11 is prime
>>13 is prime
…
>> 89 is prime
>> 97 is prime
```



Rangkuman

- Struktur pembuatan keputusan mengharuskan program untuk menentukan satu atau lebih kondisi yang akan dievaluasi atau diuji oleh program.MATLAB menyediakan beberapa fungsi struktur keputusan antara lain :
  - if ... end statement
  - if...else...end statement
  - If... elseif...elseif...else...end statements
  - nested if statements

- Struktur perulangan mengijinkan statement suatu program untukdieksekusi secara berulang-ulang. Matlab memiliki beberapa fungsi perintah untuk melakukan proses pengulangan yaitu :
  - while loop
  - for loop
  - nested loops

Tugas

1. Pahami setiap perintah dan Lakukan praktek pada komputer, semua tutorial diatas.
2. Buatlah latihan sendiri dengan mengubah-ubah tutorial diatas

Test Formatif



KEGIATAN 5

1.14 Persamaan Aljabar Dasar MATLAB

Perintah solve digunakan untuk memecahkan permasalahan. Bentuk sederhananya, fungsi solve meletakkan persamaan dalam tanda kurung dengan diapit oleh tanda petik dua sebagai argumennya.

Sebagai contoh, memcari nilai x dalam persamaan x-5 = 0

Buatlah skrip seperti dibawah ini :

```
solve('x-5=0')
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

       5
```

Contoh lain:

Buatlah skrip seperti dibawah ini :

```
y = solve('x-5 = 0')
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>y =

       5
```

Contoh lain :

Buatlah skrip seperti dibawah ini :

```
solve('x-5')
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

       5
```



Jika akan memecahkan persamaan dengan banyak variable, perintah solve ditulis dalam bentuk :

```
solve(equation, variable)
```

Sebagai contoh, mencari nilai nilai v dari persamaan v – u – 3t2 = 0 :

Buatlah skrip seperti dibawah ini :

```
solve('v-u-3*t^2=0', 'v')
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =
 3*t^2 + u
```

1.14.1 Pemecahan Persamaan Aljabar Dasar Octave

Perintah roots digunakan untuk memecahkan persamaan aljabar dalam Octave. Sebagai contoh, mencari x dalam persamaan x-5 = 0

Buatlah skrip seperti dibawah ini :

```
roots([1, -5])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =
 5
```

Contoh lain :

Buatlah skrip seperti dibawah ini :

```
y = roots([1, -5])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>y =
 5
```



1.14.2 Pemencahan Persamaan Kuadrat MATLAB

Perintahsolve dapat juga untuk memecahkan persamaan orde tinggi. Fungsi solve akan menghasilkan akar-akar persamaan dalam bentuk array.

Sebagai contoh, memecahkan persamaan x^2 -7x +12 = 0.

Buatlah skrip seperti dibawah ini :

```
eq = 'x^2 -7*x + 12 = 0';

s = solve(eq);

disp('The first root is: '), disp(s(1));

disp('The second root is: '), disp(s(2));
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>The first root is:

3

>>The second root is:

4
```

1.14.3 Pemecahan Persamaan Kuadrat Octave

Sebagai contoh, memecahkan persamaan kuadrat x^2 -7x +12 = 0 dalam Octave.

Buatlah skrip seperti dibawah ini :

```
s = roots([1, -7, 12]);

disp('The first root is: '), disp(s(1));

disp('The second root is: '), disp(s(2));
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
The first root is:

 4

The second root is:

 3
```



1.14.4 Pemecahan Persamaan Orde Tinggi MATLAB

Perintah solve dapat juga memecahkan persamaan kuadrat orde tinggi.

Sebagai contoh, memecahkan persamaan $(x-3)^2(x-7) = 0$

Buatlah skrip seperti dibawah ini :

```
solve('(x-3)^2*(x-7)=0')
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

  3

  3

  7
```

Dalam kasus persamaan orde tinggi, akar-akar persamaan terdiri dari beberapa bilangan riel dan imajinir.

Sebagai contoh, memecahkan persamaan orde empat $x^4 - 7x^3 + 3x^2 - 5x + 9 = 0$.

Buatlah skrip seperti dibawah ini :

```
eq = 'x^4 - 7*x^3 + 3*x^2 - 5*x + 9 = 0';

s = solve(eq);

disp('The first root is: '), disp(s(1));

disp('The second root is: '), disp(s(2));

disp('The third root is: '), disp(s(3));

disp('The fourth root is: '), disp(s(4));

disp('Numeric value of first root'), disp(double(s(1)));

disp('Numeric value of second root'), disp(double(s(2)));

disp('Numeric value of third root'), disp(double(s(3)));

disp('Numeric value of fourth root'), disp(double(s(4)));
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>The first root is:
6.630396332390718431485053218985
>>The second root is:
1.0597804633025896291682772499885
>>The third root is:
-0.345088397846654 - 1.07783629546301765*i
>> The fourth root is:
-0.345088397846654+ 1.07783629546301765*i
>>Numeric value of first root
   6.6304
>>Numeric value of second root
   1.0598
>>Numeric value of third root
  -0.3451 - 1.0778i
>>Numeric value of fourth root
  -0.3451 + 1.0778i
```

1.14.5 Pemecahan Persamaan Orde Tinggi Octave

Sebagai contoh, memecahkan persamaan orde empat :

$$x^4 - 7x^3 + 3x^2 - 5x + 9 = 0.$$

Buatlah skrip seperti dibawah ini :

```
v = [1, -7,  3, -5, 9];
s = roots(v);
% converting the roots to double type
disp('Numeric value of first root'), disp(double(s(1)));
disp('Numeric value of second root'), disp(double(s(2)));
disp('Numeric value of third root'), disp(double(s(3)));
disp('Numeric value of fourth root'), disp(double(s(4)));
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>Numeric value of first root

 6.6304

>>Numeric value of second root

-0.34509 + 1.07784i

>>Numeric value of third root

-0.34509 - 1.07784i

>>Numeric value of fourth root

 1.0598
```

1.14.6 Pemecahan Persamaan Sistem MATLAB

Perintah solve dapat juga digunakan untuk menberikan solusi dari persamaan system yang mengandung banyak.

Sebagai contoh, memecahkan persamaan-persamaan dibawah ini :

$5x + 9y = 5$

$3x – 6y = 4$

Buatlah skrip seperti dibawah ini :

```
s = solve('5*x + 9*y = 5','3*x - 6*y = 4');

s.x

s.y
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

 22/19

>>ans =

-5/57
```



Contoh yang lain :

x + 3y -2z = 5

3x + 5y + 6z = 7

2x + 4y + 3z = 8

1.14.7 Pemecahan Persamaan Sistem Octave

Sebagai contoh, menyelesaikan persamaan-persamaan dibawah ini :

5x + 9y = 5

3x – 6y = 4

Persamaan system linier seperti ini dapat ditulis sebagai persamaan matriks tunggal Ax = b, dimanaAadalah koefisien matriks, b adalah vector kolom yang teridir dari sisi kanan dari persamaan linier dan x adalah vector kolom yang menampilkan hasil.

Buatlah skrip seperti dibawah ini :

```
A = [5, 9; 3, -6];

b = [5;4];

A \ b
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

   1.157895

  -0.087719
```

Contoh yang lain :

x + 3y -2z = 5

3x + 5y + 6z = 7

2x + 4y + 3z = 8



1.14.8 Menguraikan dan Menyatukan Persamaan-persamaan MATLAB

Perintah expand dan perintah masing-masing adalah menguraikan dan penyederhanakan persamaan-persamaan.

Bila bekerja dengan symbol-simbol, variable-variabel didefenisikan terlebih dahulu sebagai symbol.

Buatlah skrip seperti dibawah ini :

```
syms x %symbolic variable x
syms y %symbolic variable x
% expanding equations
expand((x-5)*(x+9))
expand((x+2)*(x-3)*(x-5)*(x+7))
expand(sin(2*x))
expand(cos(x+y))
% collecting equations
collect(x^3 *(x-7))
collect(x^4*(x-3)*(x-5))
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =
x^2 + 4*x - 45
>>ans =
x^4 + x^3 - 43*x^2 + 23*x + 210
>>ans =
 2*cos(x)*sin(x)
>>ans =
cos(x)*cos(y) - sin(x)*sin(y)
>>ans =
x^4 - 7*x^3
>>ans =
x^6 - 8*x^5 + 15*x^4
```



1.14.9 Menguraikan dan Menyatukan Persamaan-persamaan Octave

Perintah expand dan perintah masing-masing adalah menguraikan dan penyederhanakan persamaan-persamaan.

Bila bekerja dengan symbol-simbol, variable-variabel didefenisikan terlebih dahulu sebagai symbol, namun dalam octave, penentuan symbol mempunyai pendekatan yang berbeda.

Buatlah skrip seperti dibawah ini :

```
% first of all load the package, make sure its installed.
pkg load symbolic
% make symbols module available
symbols
% define symbolic variables
x = sym ('x');
y = sym ('y');
z = sym ('z');
% expanding equations
expand((x-5)*(x+9))
expand((x+2)*(x-3)*(x-5)*(x+7))
expand(Sin(2*x))
expand(Cos(x+y))
 % collecting equations
collect(x^3 *(x-7), z)
collect(x^4*(x-3)*(x-5), z)
```



Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

-45.0+x^2+(4.0)*x

>>ans =

210.0+x^4-(43.0)*x^2+x^3+(23.0)*x

>>ans =

sin((2.0)*x)

>>ans =

cos(y+x)

>>ans =

x^(3.0)*(-7.0+x)

>>ans =

(-3.0+x)*x^(4.0)*(-5.0+x)
```

1.14.10 Faktorisasi dan Penyederhanaan Persamaan Aljabar

Perintah factor adalah memfaktorkan sebuah persamaan dan perintah simplify adalah menyederhanakan sebuah persamaan.

Contoh :Buatlah skrip seperti dibawah ini :

```
syms x

syms y

factor(x^3 - y^3)

factor([x^2-y^2,x^3+y^3])

simplify((x^4-16)/(x^2-4))
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

```
>>ans =

(x - y)*(x^2 + x*y + y^2)

>>ans =

[ (x - y)*(x + y), (x + y)*(x^2 - x*y + y^2)]

>>ans =

x^2 + 4
```

1.15 Menggambar MATLAB

Untuk menggambar sebuah persamaan, diperlukan langkah-langkah berikut ini :

- Menentukan x, oleh kisaran tertentu dari nilai x dimana fungsi akan digambarkan
- Menentukan y = f(x)
- Memanggil plot untuk gambar fungsi f(x)

Sebagai contoh, menggambarkan fungsi sederhana y = x untuk kisaran x dari 0 sampai 100 dengan langkah 5.

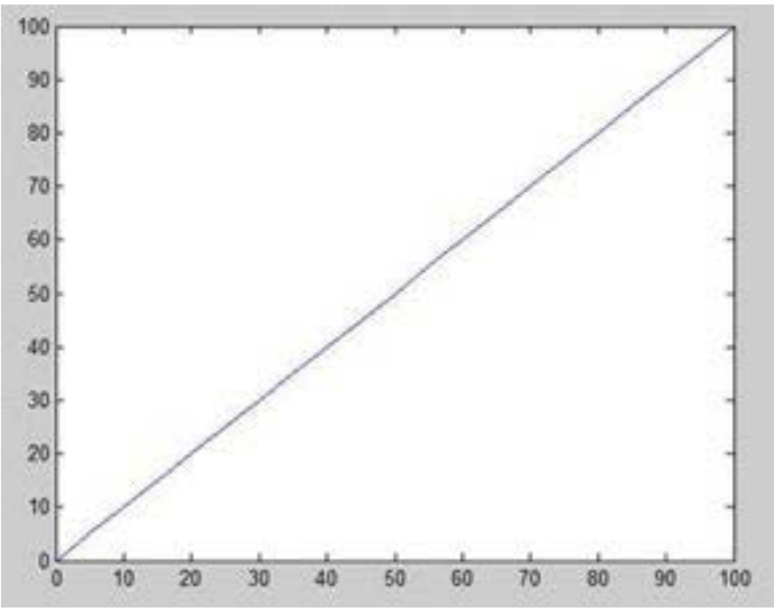
Buatlah skrip seperti dibawah ini :

```
x = [0:5:100];

y = x;

plot(x, y)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 1. 8Jendela utama MATLAB.



Contoh lain, menggambarkan fungsi $y = x^2$. Dalam contoh ini akan digambarkan dua gambar untuk fungsi yang sama, tetapi dalam langkah kisaran yang berbeda yaitu langkah 20 dan 5.

Buatlah skrip seperti dibawah ini :

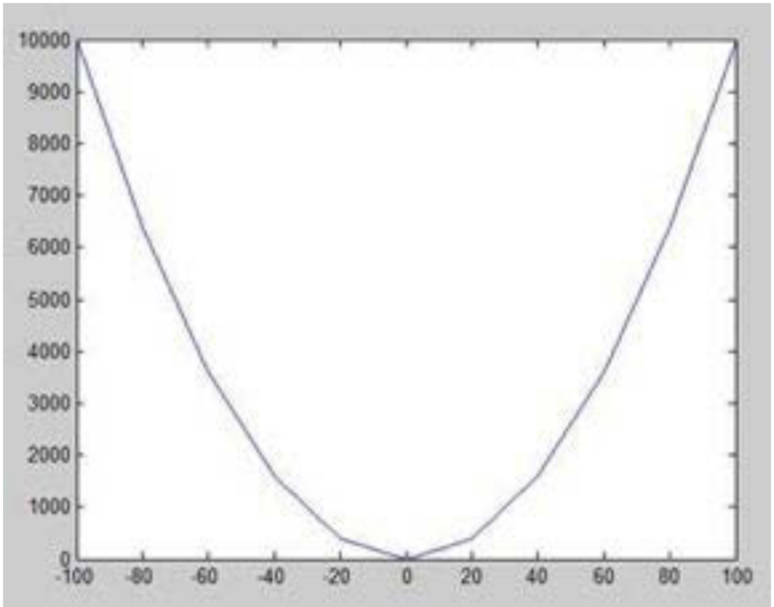
```
x = [1 2 3 4 5 6 7 8 9 10];

x = [-100:20:100];

y = x.^2;

plot(x, y)
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 1. 9Jendela utama MATLAB.

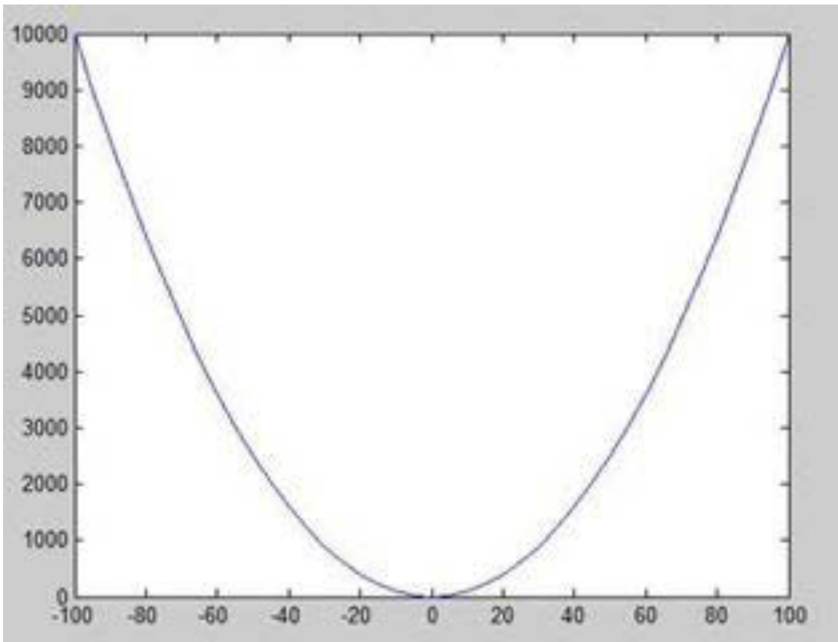
Ganti kode, dengan mengurangi langkah menjadi 5:

```
x = [-100:5:100];

y = x.^2;

plot(x, y)
```

MATLAB draws a smoother graph:



Gambar 1. 10Jendela utama MATLAB.



1.15.1 Adding Title, Labels, Grid Lines and Scaling on the Graph

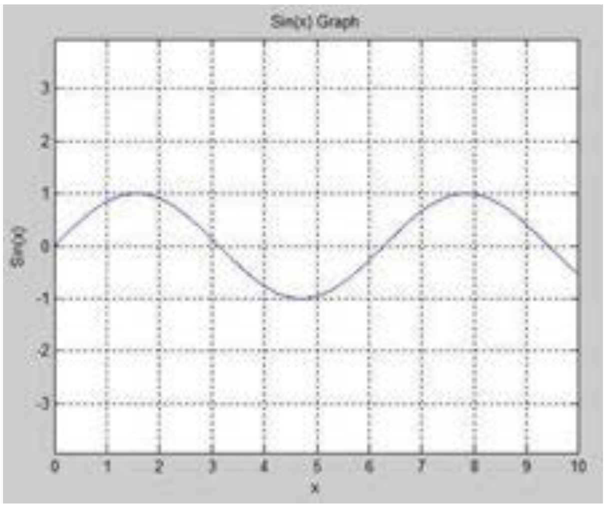
MATLAB mengijikan untuk menambah judul, label x, label y, garis grid dan juga mengatur aksis x,y untuk gambar.

- Perintah xlabel and ylabel menghasilkan label sepanjang aksis x dan aksis y.
- Perintah title mengijinkan untuk meletakkan judul pada gambar.
- Perintah grid on mengijinkan untuk meletakan grid pada gambar.
- Perintah axis equal mengijinkan menghasilkan gambar dalam factor skala dan langkah yang samauntuk kedua aksis.
- Perintah axis square menghasilkan sebuah gambar kotak.

Buatlah skrip seperti dibawah ini :

```
x = [0:0.01:10];
y = sin(x);
plot(x, y), xlabel('x'), ylabel('Sin(x)'), title('Sin(x) Graph'),
grid on, axis equal
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1.11Jendela utama MATLAB.



1.15.2 Drawing Multiple Functions on the Same Graph

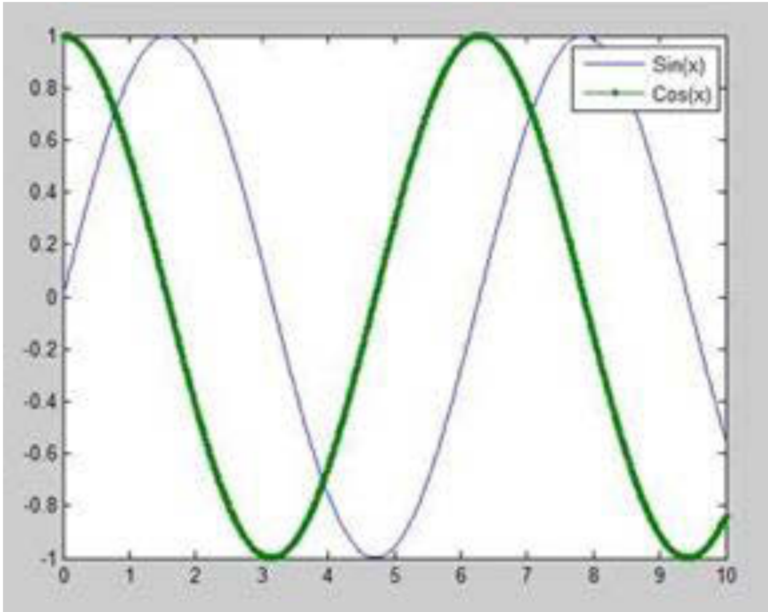
Anda dapat menggambarkan multi gambar pada gambar yang sama.

Contoh :

Buatlah skrip seperti dibawah ini :

```
x = [0 : 0.01: 10];
y = sin(x);
g = cos(x);
plot(x, y, x, g, '.-'), legend('Sin(x)', 'Cos(x)')
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 12Jendela utama MATLAB.



1.15.3 Penentuan Warnapada Grafik

MATLAB menyediakan delapan opsi warna dasar untuk menggambar grafik :

Color	Code
White	W
Black	K
Blue	B
Red	R
Cyan	C
Green	G
Magenta	M
Yellow	Y

Contoh :

$f(x) = 3x^4 + 2x^3 + 7x^2 + 2x + 9$ dan

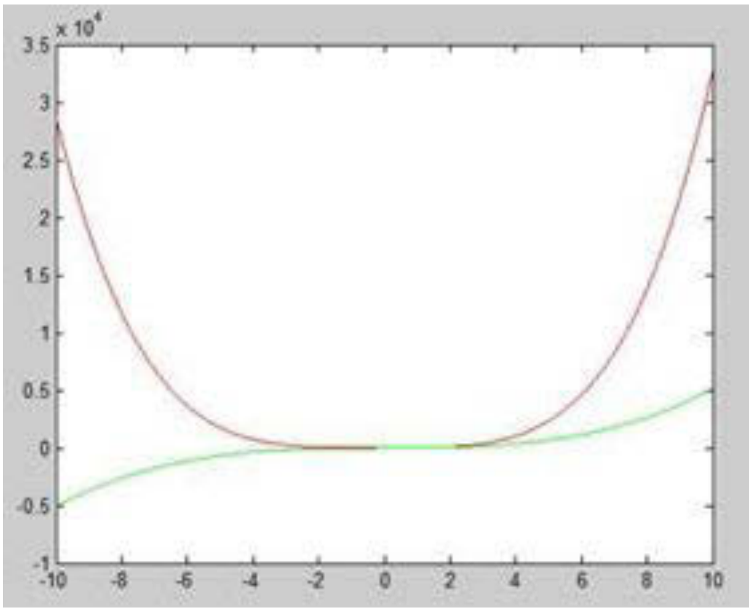
$g(x) = 5x^3 + 9x + 2$

Buatlah skrip seperti dibawah ini :

```
x = [-10 : 0.01: 10];
y = 3*x.^4 + 2 * x.^3 + 7 * x.^2 + 2 * x + 9;
g = 5 * x.^3 + 9 * x + 2;
plot(x, y, 'r', x, g, 'g')
```



Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 13Jendela utama MATLAB.

1.15.4 Penentuan Skala Aksis

Perintah axis mengijinkan untuk menentukan skala aksis.Disediakan nilai minimum dan maksimum untuk x dan y menggunakan perintah axis dengan cara berikut :

```
axis ( [xmin xmax ymin ymax] )
```

Contoh :

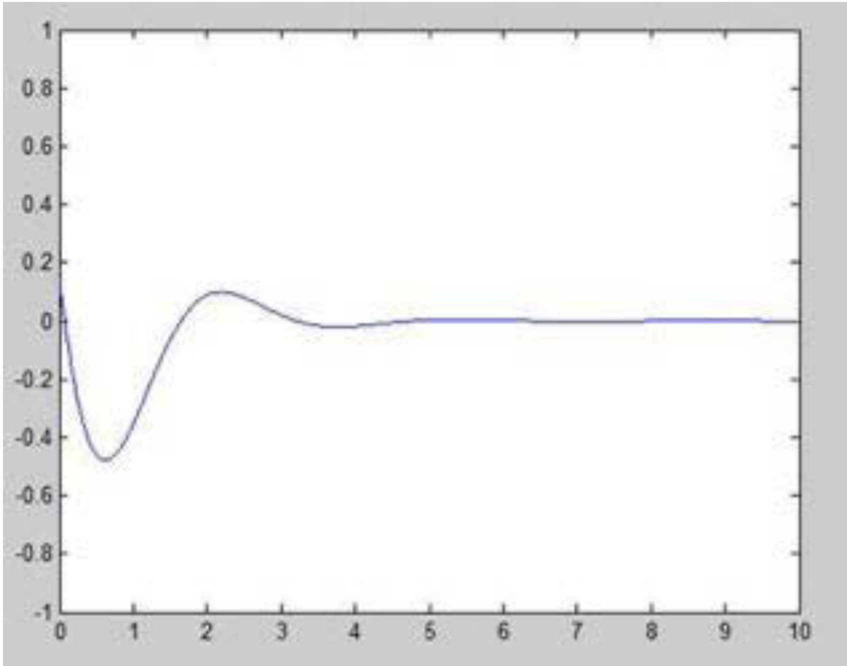
Buatlah skrip seperti dibawah ini :

```
x = [0 : 0.01: 10];

y = exp(-x).* sin(2*x + 3);

plot(x, y), axis([0 10 -1 1])
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 14Jendela utama MATLAB.

1.15.5 Membuat Sub-Gambar

Jika Anda membuat sebuah array dari gambar dalam figur yang sama, setiap gambar dipanggil sebagai sub-gambar. Perintah subplot adalah untuk membuat sub-gambar.

Sintaks :

```
subplot(m, n, p)
```

dimana, m dan n adalah jumlah baris dan kolom dari array gambar dan p menentukan dimana akan meletakkan gambar tersebut.

Masing-masing gambar dibuat dengan perintah subplot.

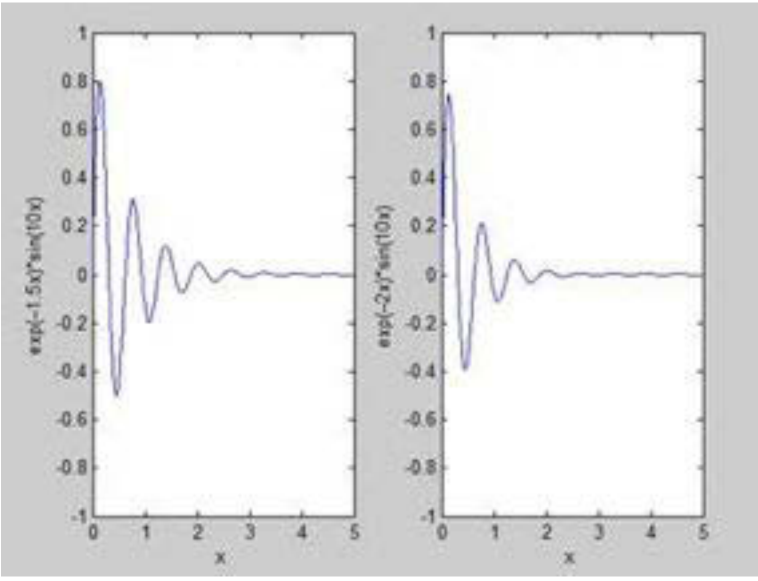


Contoh :

Buatlah skrip seperti dibawah ini :

```
x = [0:0.01:5];
y = exp(-1.5*x).*sin(10*x);
subplot(1,2,1)
plot(x,y), xlabel('x'),ylabel('exp(–1.5x)*sin(10x)'),axis([0 5 -1 1])
y = exp(-2*x).*sin(10*x);
subplot(1,2,2)
plot(x,y),xlabel('x'),ylabel('exp(–2x)*sin(10x)'),axis([0 5 -1 1])
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 15Jendela utama MATLAB.



1.16 Grafik MATLAB

Sub bab ini akan menjelaskan lebih lanjut tentang kemampuan menggambar grafik MATLAB :

- Menggambar chart bar
- Menggambarkontur
- Menggambar tiga dimensi

1.16.1 Menggambar Chart Bar

Perintah bar menggambar chart dua dimensi.

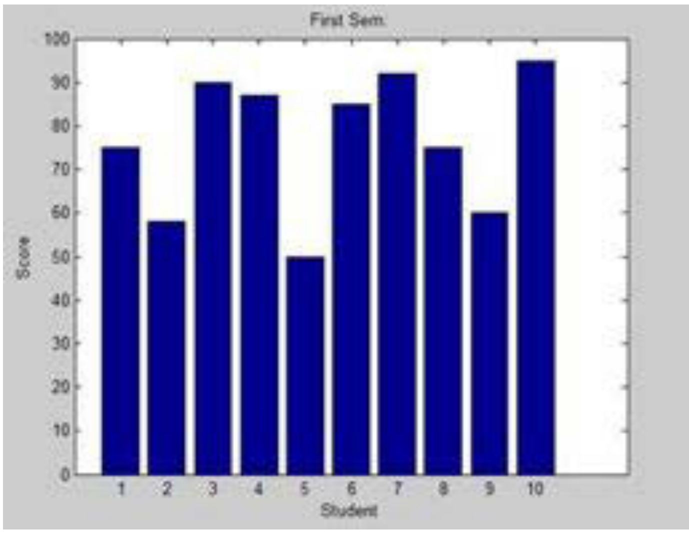
Contoh :

Nilai dari sepuluh orang dalam kelas adalah 75, 58, 90, 87, 50, 85, 92, 75, 60 dan 95. Anda akan menggambarkan chart bar dari data-data ini :

Buatlah skrip seperti dibawah ini :

```
x = [1:10];
y = [75, 58, 90, 87, 50, 85, 92, 75, 60, 95];
bar(x,y), xlabel('Student'),ylabel('Score'),
title('First Sem:')
print -deps graph.eps
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 16Jendela utama MATLAB.



1.16.2 MenggambarkanKontur

Sebuah garis kontur dari sebuah fungsi dua variable adalah sebuah kurva sepanjang fungsi mempuyai nilai. Garis kontur digunakan untuk membuat peta kontur dengan menghubungkan titik-titik elevasi yang sama, seperti rata-rata permukaan air laut.

Contoh :

Membuah sebuah peta kontur yang menunjukkan garis kontur untuk fungsi g = f(x, y). Fungsi ini mempunyai dua variable. Jadi, kita akan membuat dua variable bebas yaitu dua himpunan data x dan y. Ini dilakukan oleh perintah meshgrid.

Perintah meshgrid digunakan untuk membuat sebuah matriks elemen-elemen yang memberikan kisaran x dan y dengan spesifikasi masing-masing.

Kita gambar fungsi g = f(x, y), dimana $-5 \le x \le 5$, $-3 \le y \le 3$. Kita berikan penambahan langkah 0.1 untuk kedua nilai.Variabel x dan y ditentukan sebagai :

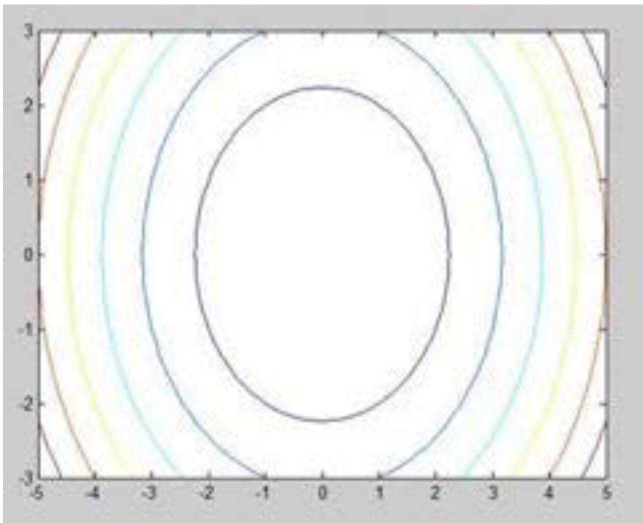
[x,y] = meshgrid(–5:0.1:5, –3:0.1:3); Kita tentukan fungsi sebagai : $x^2 + y^2$

Buatlah skrip seperti dibawah ini :

```
[x,y] = meshgrid(-5:0.1:5,-3:0.1:3); %independent variables
g = x.^2 + y.^2;                   % our function
contour(x,y,g)                     % call the contour function
print -deps graph.eps
```



Setelah dijalankan, MATLAB akan menampilkan hasil :

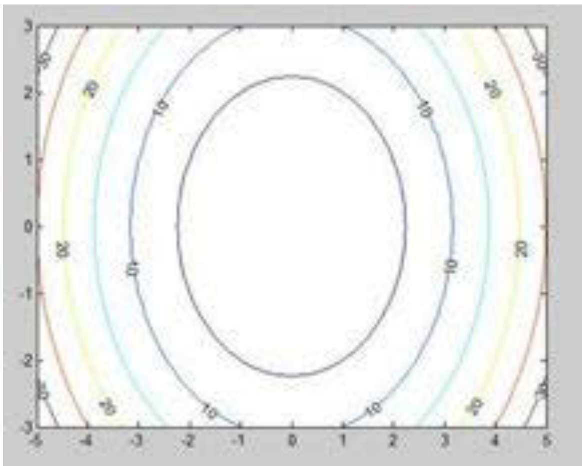


Gambar 1. 17Jendela utama MATLAB.

Kita modifikasi, skrip seperti dibawah ini :

```
[x,y] = meshgrid(-5:0.1:5,-3:0.1:3); %independent variables
g = x.^2 + y.^2;                    % our function
[C, h] = contour(x,y,g);            % call the contour function
set(h,'ShowText','on','TextStep',get(h,'LevelStep')*2)
print -deps graph.eps
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 18 Jendela utama MATLAB.



1.16.3 Gambar Tiga Dimensi

Gambar tiga dimensi pada dasarnya menampilkan sebuah permukaan yang ditentukan oleh dua variable, g = f (x,y).

Sperti sebelumnya, untuk menentukan fungsi g, pertama kita membuat sebuah himpunan dari titik-titik (x,y) pada domain fungsi yang menggunakan perintah meshgrid. Selanjutnya, kita menetapkan fungsi itu sendiri.Akhirnya, kita menggunakan perintah surf untuk menggambar sebuah gambar surface.

Contoh menggambar peta permukaan tiga dimensi untuk fungsi g = xe-(x^2 + y^2)

Buatlah skrip seperti dibawah ini :

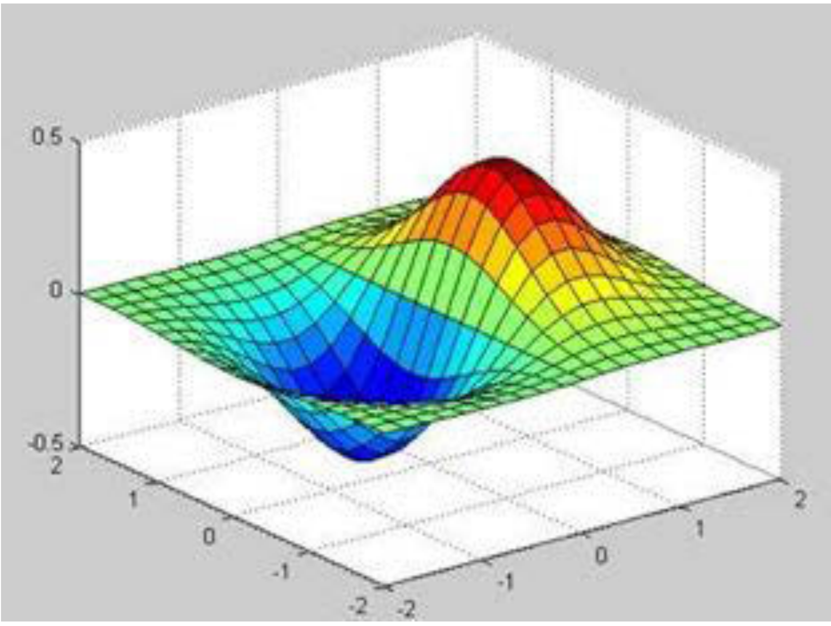
```
[x,y] = meshgrid(-2:.2:2);

g = x .* exp(-x.^2 - y.^2);

surf(x, y, g)

print -deps graph.eps
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 1. 19 Jendela utama MATLAB.



Rangkuman

- Persamaan aljabar dasar dapat dipecahkan secara mudah dengan menggunakan aplikasi MATLAB. Berbagai contoh persamaan aljabar seperti persamaan garis lurus, persamaan akar kuadrat, persamaan orde satu, persamaan orde tinggi dan lain-lain dapat mudah dipecahkan dengan menggunakan perintah “solve” dan “roots”.

- Fungsi gambar pada MATLAB berguna untuk merepresentasikan hasil dari suatu persamaan, untuk menggambar hasil suatu persamaan diperlukan langkah-langkah berikut ini :
  - Menentukan x, oleh kisaran tertentu dari nilai x dimana fungsi akan digambarkan
  - Menentukan y = f(x)
  - Memanggil plot untuk gambar fungsi f(x).

- Syntaks perintah gambar pada MATLABterdiri dari beberapa perintah seperti berikut :
  - Menggambar persamaan dua dimensi : Plot (x,y)
  - Menggambar chart bar : bar (x,y)
  - Menggambarkontur :Contour (x,y,g) dimana g=f(x)
  - Menggambar tiga dimensi : surf (x,y,g) dimana g=f(x)



Tugas

1. Pahami setiap perintah dan Lakukan praktek pada komputer, semua tutorial diatas.
2. Buatlah latihan sendiri dengan mengubah-ubah tutorial diatas

Tes Formatif



KEGIATAN BELAJAR 2

Sebelum proses pembelajaran di kelas berlangsung, sebaiknya siswa mempersiapkan diri dengan belajar mandiri sesuai dengan urutan materi yang akan diberikan.Sebagai gambaran kegiatan belajar siswa seperti pada tabel berikut :

NO	KEGIATAN SISWA	KETERANGAN
1	Persiapan Kegiatan 1 1. Siswa membaca materi pendahuluan 2. Siswa mempelajari materi identifikasi .... 3. Siswa mempelajari .... 4. Siswa mencoba mengerjakan soal tes formatif 1	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 1
2	Persiapan Kegiatan 2 5. Siswa membaca materi ...... 6. Siswa mempelajari materi .... 7. Siswa mempelajari .... 8. Siswa mencoba mengerjakan soal tes formatif 2	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 2
3	Persiapan Kegiatan 3 4. Siswa mempelajari materi ...... 5. Siswa mempelajari ...... 6. Siswa mencoba mengerjakan soal tes formatif 3	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 3



Selanjutnya siswa mendengarkan penyampaian materi pembelajaran di setiap pertemuan oleh guru serta menyesuaikan dengan model pembelajaran yang digunakan. Misalnya saatnya harus aktif mengerjakan soal maupun praktikum, maka siswa juga harus aktif dan kreatif. Melalui langkah kegiatan pembelajaran yang saling melengkapi diharapkan siswa dapat mencapai kompetensi yang distandarkan.

A. Tujuan Pembelajaran

Setelah mempelajari materi tentang dasar teknik kontrol, diharapkan siswa dapat:

1. mengidentifikasi ....
2. mengidentifikasi ....

B. Uraian Materi

- Dasar sistem kendali Mikrokontroller, komponen dan spesifikasinya serta perbandingan sistem kendali Mikrokontroller dengan sistem kendali yang lain.
- Teknik pemrograman Mikrokontroller.
- Teknik pemasangan dan pengawatan peralatan input output.
- Penggunaan alat pemrogram dengan komputer yang dilengkapi dengan software ladder
- Pengoperasian sistem kendali Mikrokontroller

C. Alokasi Waktu

4 jam pelajaran

D. Metode Pembelajaran

Teori dan Praktek

E. Media pembelajaran

- PC/Notebook
- Windows 7
- Livewire



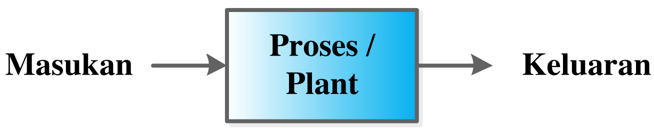
KEGIATAN 1

2.1 Pengertian Kontrol

Teknik kontrol berkaitan dengan pemahaman dan pengontrolan bahan dan kekuatan alam untuk kepentingan umat manusia . Tujuan ganda pemahaman dan kontrolan saling melengkapi karena sistem kontrol yang efektif mensyaratkan bahwa sistem dapat dipahami dan dimodelkan . Selain itu , teknik kontrol harus mempertimbangkan kontrolan sistem yang sulit dipahami seperti sistem proses kimia . Tantangan saat ini untuk kontrolan adalah pemodelan dan kontrolan modern, kompleks, sistem yang saling terkait seperti sistem kontrol lalu lintas , proses kimia , dan sistem robot .

Teknik kontrol didasarkan pada dasar-dasar teori umpan balik dan analisis sistem linear, dan mengintegrasikan konsep-konsep teori jaringan dan teori komunikasi. Oleh karena itu teknik kontrol tidak terbatas pada disiplin ilmu saja tapi berlaku juga untuk teknik aeronautika, kimia, mekanik, lingkungan, sipil, dan listrik.

Sebuah sistem kontrol adalah sebuah interkoneksi dari komponen membentuk konfigurasi sistem yang akan memberikan respon sistem yang diinginkan. Dasar analisis sistem adalah dasar yang disediakan oleh teori sistem linear, yang mengasumsikan hubungan sebab-akibat untuk komponen sistem. Oleh karena itu komponen atau proses yang dikontrolkan dapat diwakili oleh blok, seperti yang ditunjukkan pada Gambar 2.1. Hubungan input-output merupakan hubungan sebab-akibat dari proses, yang pada gilirannya merupakan pengolahan sinyal input untuk memberikan variabel sinyal output, sering dengan amplifikasi daya.



Gambar 2.1 Proses yang akan dikontrol.

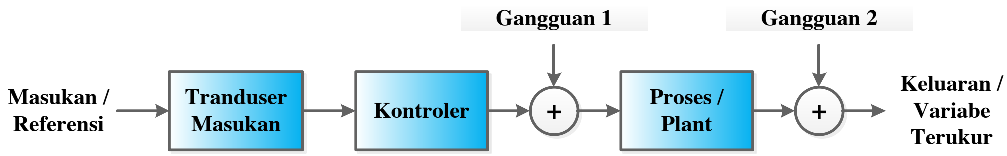
Sistem kontrol adalah suatu sistem yang bertujuan untuk mengendalikan suatu proses agar keluaran yang dihasilkan dapat dikontrolkan sehingga tidak terjadi kesalahan terhadap referensi yang ditentukan. Dalam hal ini keluaran yang



dikontrolkan adalah kestabilan, ketelitian dan kedinamisannya. Secara umum sistem kontrol dapat dibagi menjadi 2 jenis yaitu sistem kontrol rangkaian terbuka dan rangkaian tertutup.

2.1.1 Sistem Kontrol Rangkaian Terbuka

Sebuah sistem rangkaian terbuka umum ditunjukkan pada Gambar 2.2. Dimulai dengan subsistem yang disebut transducer input, yang mengubah bentuk input yang digunakan oleh controller. Kontroler mengendalikan proses atau plant. Input kadang-kadang disebut referensi, sedangkan output dapat disebut variabel yang dikontrol. Sinyal lain, seperti gangguan, akan ditampilkan ditambahkan ke kontroler dan output proses melalui penjumlah, yang menghasilkan jumlah aljabar sinyal input dengan menggunakan tanda-tanda yang terkait.



Gambar 2.2 Sistem kontrol terbuka (tanpa umpan balik)

Karakteristik yang membedakan dari sistem rangkaian terbuka adalah bahwa ia tidak dapat mengkompensasi Gangguan 1 yang menambah sinyal mengemudi kontroler Gambar 2.2 . Sebagai contoh, jika kontroler adalah sebuah penguat elektronik dan Gangguan 1 adalah kebisingan, maka sinyal kebisingan menambah sinyal penguat yang akan juga mengendalikan proses, merusak output dengan efek bising suara. Output dari sistem rangkaian terbuka rusak tidak hanya oleh sinyal gangguan yang menambah sinyal kontroler , tetapi juga oleh gangguan pada output (Gangguan 2 pada Gambar 2.2).



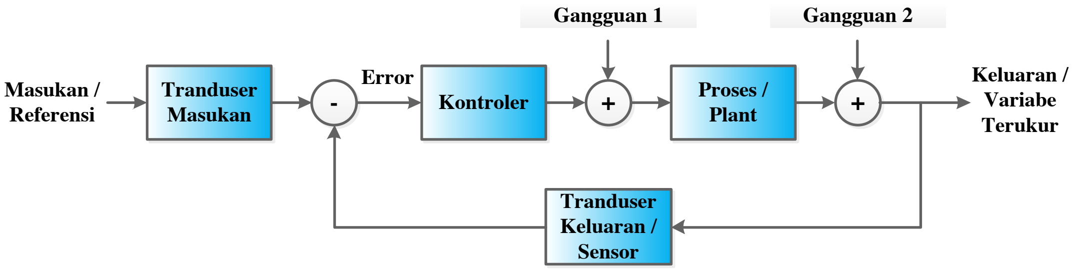
2.1.2 Sistem Kontrol Rangkaian Tertutup

Kelemahan dari sistem rangkaian terbuka, yaitu kepekaan terhadap gangguan dan ketidakmampuan untuk mengoreksi gangguan ini, dapat diatasi dalam sistem rangkaian tertutup. Arsitektur umum dari sistem rangkaian tertutup ditunjukkan pada Gambar 2.3 . Input transduser mengubah bentuk input ke bentuk yang digunakan oleh kontroler. Sebuah transduser output, atau sensor, mengukur respon output dan mengubahnya menjadi bentuk yang digunakan oleh kontroler. Sebagai contoh, jika controller menggunakan sinyal listrik untuk mengoperasikan katup sistem kontrol suhu, kondisi input dan output temperatur diubah menjadi sinyal listrik. Kondisi input dapat diubah menjadi tegangan oleh potensiometer, resistor variabel, dan suhu output dapat diubah menjadi tegangan oleh thermistor, sebuah perangkat yang hambatan listrik berubah dengan suhu. Pertama penjumlah aljabar menambahkan sinyal dari input ke sinyal dari output, yang datang melalui jalur umpan balik, jalur kembali dari output ke persimpangan penjumlahan. Dalam Gambar 2.3, sinyal input dikurangi oleh sinyal output. Hasilnya umumnya disebut sinyal penggerak. Namun, dalam sistem di mana kedua input dan output transduser memiliki penguatan satu, nilai sinyal penggerak adalah sama dengan perbedaan yang sebenarnya antara input dan output. Dalam kondisi ini, sinyal penggerak disebut kesalahan.

Sistem rangkaian tertutup mengkompensasi gangguan dengan mengukur output respon, hasil pengukuran yang kembali melalui jalur umpan balik, dan membandingkan dengan respon masukan di persimpangan penjumlahan. Jika ada perbedaan antara dua tanggapan, sistem mendorong plant, maka sinyal penggerak, akan mengoreksi. Jika tidak ada perbedaan, sistem tidak mendorong plant, karena respon plant sudah sama dengan respon yang diinginkan. Sistem rangkaian tertutup, memiliki keuntungan akurasi yang lebih besar dibandingkan dengan sistem rangkaian terbuka. Mereka peka terhadap kebisingan, gangguan, dan perubahan lingkungan. Respon dan error steady-state dapat dikontrol lebih mudah dan dengan fleksibilitas yang lebih besar dalam sistem rangkaian tertutup, dengan sederhana penyesuaian penguatan dalam rangkaian dan kadang-kadang dengan mendesain ulang kontroler. Di sisi lain, sistem rangkaian tertutup lebih kompleks dan mahal dibandingkan dengan sistem rangkaian terbuka.



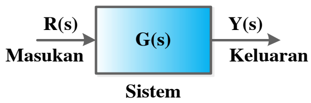
Singkatnya, sistem yang melakukan pengukuran hasil sebelumnya dan koreksi disebut rangkaian tertutup atau kontrol umpan balik. Sistem yang tidak memiliki properti pengukuran dan koreksi disebut sistem rangkaian terbuka.



Gambar 2.3 Fungsi alih sistem kontrol rangkaian tertutup

2.1.3 Fungsi Alih Sistem Kontrol

Subsistem direpresentasikan sebagai blok dengan input, output, dan fungsi alih. Ketika beberapa subsistem yang saling berhubungan, beberapa elemen skema harus ditambahkan ke diagram blok. Semua bagian komponen diagram blok untuk sistem linear waktu - invariant ditunjukkan pada Gambar 2.4.

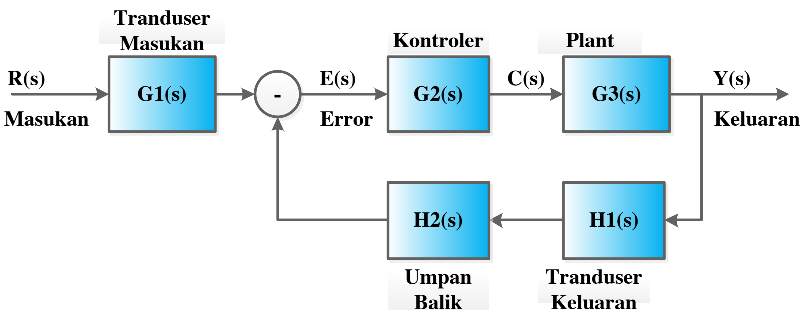


Gambar 2.4 Fungsi alih sistem kontrol rangkaian terbuka

Fungsi alih sistem kontrol rangkaian terbuka adalah :

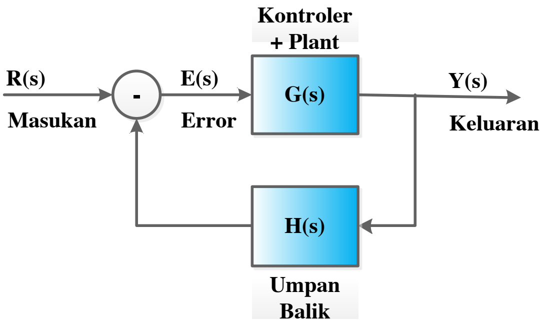
$$Y(s)/R(s) = G(s)$$

Sistem umpan balik membentuk dasar untuk studi rekayasa kontrol sistem. Sistem umpan balik yang khas, ditunjukkan pada Gambar 2.5 (a), sebuah model sederhana ditunjukkan pada Gambar 2.5 (b).

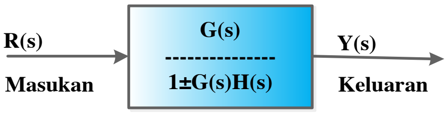


(a)





(b)



(c)

Gambar 2.5 Fungsi alih sistem kontrol rangkaian tertutup

E(s) = R(s)±Y(s)H(s))

$$E(s) = \frac{Y(s)}{G(s)}$$

Fungsi alih sistem kontrol rangkaian tertutup adalah :

$$Ge(s) = \frac{G(s)}{1 \pm G(s)H(s)}$$

Mengarahkan perhatian kita pada model yang disederhanakan, dan pemecahan untuk fungsi alih, Y(s)/R(s)=G(s), diperoleh fungsi alih yang ditunjukkan pada Gambar 2.5(c), hasilG(s)H(s) disebut fungsi alih rangkaian terbuka, atau penguat rangkaian.

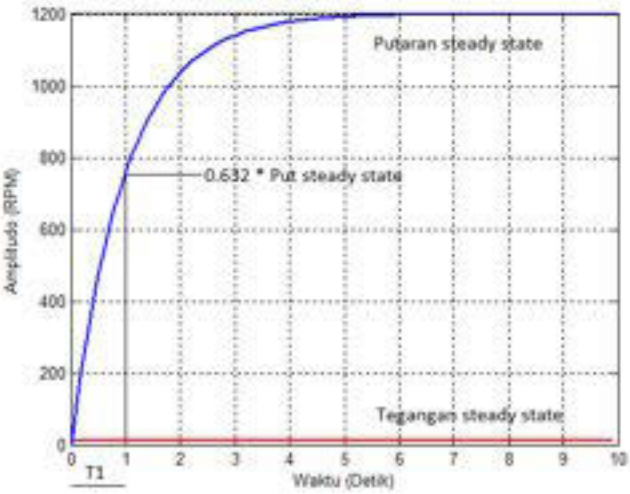


2.1.4 Model Matematika Kecepatan Putaran Motor DC orde 1

Identifikasi plant ditujukan untuk mendapatkan model matematis berupa fungsi alih yang digunakan untuk proses perancangan kontroler nantinya. Untuk jenis plant yang dibahas berupa motor DC sistem orde satu. Orde sistem menentukan jenis kontroler yang akan dipakai dan mencari nilai parameter kontroler untuk hasil respon yang diinginkan.

Fungsi alih motor DC dapat ditentukan melalui pembacaan kurva karakteristik yang didapatkan melalui pengukuran keluaran kecepatan putaran motor dan tegangan masukan motor DC, seperti yang ditunjukkan pada Gambar 2.6.

Tegangan steady state adalah tegangan masukan yang diberikan konstan pada masukan motor DC, putaran steady state adalah kecepatan putaran motor setelah mencapai putaran nominalnya setelah diberikan tegangan masukan tertentu, sementara waktu respon motor T1 adalah waktu transien yang diperlukan untuk perubahan kecepatan putaran motor mencapai putaran nominalnya.



Gambar 2.6 Kurva kecepatan putaran motor orde 1



Dari parameter kurva karakteristik tersebut diatas, didapatkan parameter model matematika orde satu :

$$K \ = \frac{yss}{xss}$$

$$T1 = t|0{,}632 * yss$$

Dimana :

yss = sinyal putaran steady state

xss = sinyal tegangan steady state

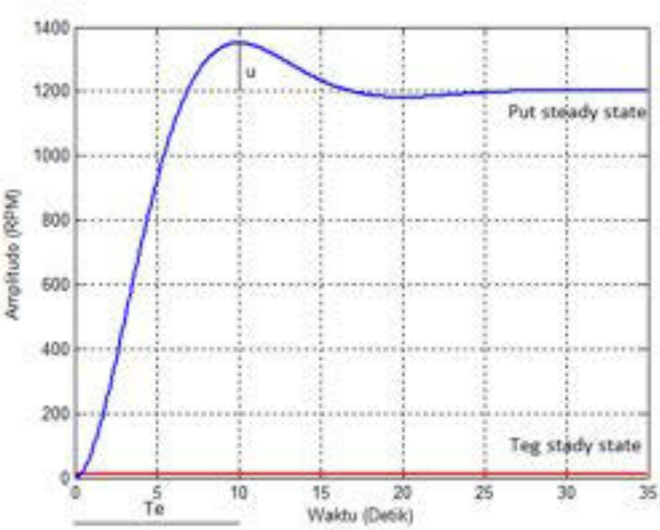
t = waktu

Fungsi alih dari kecepatan putaran motor DC orde 1 dituliskan dalam domain S seperti pada persamaan berikut :

$$G(s) = \frac{K}{T_1 s + 1}$$

2.1.5 Model Matematika Kecepatan Putaran Motor DC orde 2

Untuk jenis plant yang dibahas berupa motor DC sistem orde dua. Orde sistem menentukan jenis kontroler yang akan dipakai dan mencari nilai parameter kontroler untuk mengubah transien respon orde dua menjadi transien respon orde satu yang diinginkan.



Gambar 2.7 Kurva kecepatan putaran motor orde 2



Fungsi alih motor DC dapat ditentukan melalui pembacaan kurva karakteristik yang didapatkan melalui pengukuran keluaran kecepatan putaran motor dan tegangan masukan motor DC (Gambar 2.7.)

Tegangan steady state adalah tegangan masukan yang diberikan konstan pada masukan motor DC, putaran steady state adalah kecepatan putaran motor setelah mencapai putaran nominalnya setelah diberikan tegangan masukan tertentu, sementara waktu respon motor Te adalah waktu yang diperlukan untuk perubahan kecepatan putaran motor mencapai putaran maksimumnya dan u adalah kelebihan putaran terhadap putaran nominalnya.

Dari parameter kurva karakteristik tersebut diatas, didapatkan parameter model matematika orde dua :

$$K = \frac{yss}{xss}$$

$$\xi = \frac{-\log(u - yss)}{\sqrt{\pi^2 + \log(u - yss)^2}}$$

$$\omega = 2Te\sqrt{\frac{1 - \xi^2}{2\pi}}$$

Dimana :

yss = sinyal putaran steady state

xss = sinyal tegangan steady state

u = kelebihankecepatan putaran terhadap putaran nominalnya

Te = Waktu mencapai kecepatan putaran maksimum

Fungsi alih dari kecepatan putaran motor DC orde 2 dituliskan dalam domain S seperti pada persamaan berikut :

$$G(s) = \frac{K}{\frac{1}{\omega^2}s^2 + \frac{2\xi}{\omega}s + 1}$$



2.1.6 Kontroler PID

Kontroler PID ideal untuk domain waktu kontinyu proses SISO (single input single output) dinyatakan dalam domain Laplace sebagai berikut :

$$\mathrm{C(s) = Gc(s)E(s)}$$

Fungsi alih kontroler PID :

$$\mathrm{Gc(s) = Kp(1 + \frac{1}{Tis} + Tds)}$$

$$\mathrm{Gc(s) = \frac{C(s)}{E(s)} = Kp\frac{Ti\,Td\,s^2 + Tis + 1}{Tis}}$$

Dimana :

Kp = konstanta penguatan proposional

Ti = konstanta waktu integral

Td = konstanta waktu derivatif

Jika Ti = ∞ dan Td = 0 (yaitu kontrol P) , maka jelas bahwa nilai y rangkaian tertutup terukur akan selalu lebih kecil dari nilai r yang diinginkan (tanpa proses integrasi, ketika kesalahan positif diperlukan untuk menjaga konstan nilai terukur, dan kurang dari nilai yang diinginkan). Pengenalan tindakan integrasi memfasilitasi tercapainya kesesuaian antara nilai terukur dan nilai yang diinginkan, ketika konstanta kesalahan menghasilkan output pengontrol meningkat. Pengenalan tindakan derivatif berarti bahwa perubahan nilai yang diinginkan dapat diantisipasi, dan dengan demikian koreksi yang tepat dapat ditambahkan sebelum perubahan yang sebenarnya.

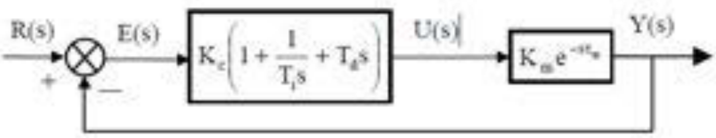
Bentuk proporsional menghasilkan nilai output yang proporsional dengan nilai kesalahan saat ini. Tanggapan proporsional dapat disesuaikan dengan mengalikan kesalahan oleh Kp konstan, yang disebut gain konstan proporsional. Sebuah penguatan proporsional yang tinggi mengakibatkan perubahan besar dalam output untuk perubahan yang diberikan dalam kesalahan. Jika penguatan proporsional terlalu tinggi, sistem dapat menjadi tidak stabil (lihat bagian lingkaran penalaan). Sebaliknya, penguatan kecil menghasilkan respon output kecil untuk kesalahan masukan yang besar, dan kontroler kurang responsif atau kurang sensitif. Jika penguatan proporsional terlalu rendah, tindakan kontrol mungkin terlalu kecil ketika menanggapi gangguan sistem. Teori Tuning dan praktek industri menunjukkan bahwa istilah proporsional harus memberikan kontribusi sebagian besar perubahan keluaran



Kontribusi dari bagian integral sebanding dengan baik besarnya maupun durasi kesalahan. Bentuk integral dalam kontroler PID adalah jumlah kesalahan sesaat dari waktu ke waktu dan memberikan akumulasi offset yang seharusnya dari sebelum diperbaiki. Akumulasi kesalahan tersebut kemudian dikalikan dengan penguatan integral (Ti) dan ditambahkan ke output kontroler. Bentuk integral mempercepat pergerakan proses menuju setpoint dan menghilangkan sisa kesalahan steady-state yang terjadi dengan kontroler proporsional murni. Namun, bila bagian integral merespon akumulasi kesalahan dari masa lalu, hal ini dapat menyebabkan nilai sekarang akan melebihi nilai setpoint.

Derivatif dari kesalahan proses dihitung dengan menentukan kemiringan kesalahan dari waktu ke waktu dan mengalikan tingkat perubahan ini dengan penguatan derivatif Kd. Besarnya kontribusi istilah derivatif untuk tindakan kontrol keseluruhan disebut penguatan derivatif, Kd. Tindakan derivatif memprediksi perilaku sistem dan dengan demikian meningkatkan settling time dan stabilitas sistem. Aksi derivatif jarang digunakan dalam praktek karena sensitivitas kebisingan yang melekat saat pengukuran. Jika kebisingan ini cukup parah, tindakan derivatif tidak akan benar-benar menentu dan menurunkan kinerja kontrol. Perubahan mendadak dalam pengukuran kesalahan (yang biasanya terjadi ketika set point berubah) menyebabkan tiba-tiba tindakan kontrol besar. Masalah ini dapat diperbaiki, jika kesalahan diukur dilewatkan melalui low-pass filter linear atau nonlinear.

Jadi, dalam bentuk yang disederhanakan, kontroler PID memungkinkan kontribusi penyelesaian dari kesalahan kontroler saat ini, saat lalu dan saat nanti.

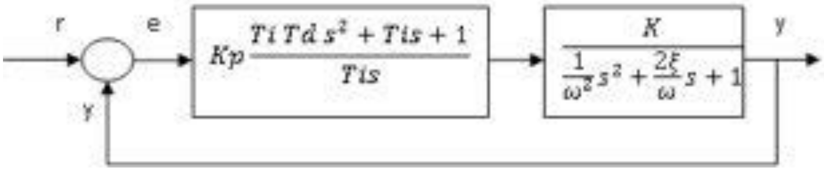


Gambar 2.8 Diagram Blok Kontrol PID



2.1.7 Disain Parameter Kontroler PID

Setelah mendapatkan model matematika kecepatan putaran motor DC dan kontroler PID, maka nilai parameter kontroler PID dapat ditentukan sebelumnya dengan penentuan transien model rangkaian tertutup dari kecepatan putaran motor DC yang diinginkan.



Gambar 2.9 Diagram blok rangkaian sistem tertutup

$$\frac{Y(s)}{R(s)} = \frac{Kp\frac{TiTds^2+Tis+1}{Tis}\frac{K}{\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1}}{1 + Kp\frac{TiTds^2+Tis+1}{Tis}\frac{K}{\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1}}$$

$$\frac{Y(s)}{R(s)} = \frac{KpK\frac{TiTds^2+Tis+1}{Tis\left(\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1\right)}}{1 + KpK\frac{TiTds^2+Tis+1}{Tis\left(\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1\right)}}$$

$$\frac{Y(s)}{R(s)} = \frac{\frac{KpK(TiTds^2+Tis+1)}{Tis\left(\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1\right)}}{\frac{Tis\left(\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1\right)+KpK(TiTds^2+Tis+1)}{Tis\left(\frac{1}{\omega^2}s^2+\frac{2\xi}{\omega}s+1\right)}}$$

$$\frac{Y(s)}{R(s)} = \frac{KpK(TiTds^2 + Tis + 1)}{Tis\left(\frac{1}{\omega^2}s^2 + \frac{2\xi}{\omega}s + 1\right) + KpK(TiTds^2 + Tis + 1)}$$

Jika $\mathrm{Ti} = \frac{2\varsigma}{\omega}$

dan $\mathrm{TiTd} = \frac{1}{\omega^2}$

maka :

$$\frac{Y(s)}{R(s)} = \frac{KpK}{Tis+KpK} = \frac{1}{\frac{Ti}{KpK}s-1}$$



Apabila model matematika rangkaian sistem tertutup adalah :

$$\frac{Y(s)}{R(s)} = \frac{1}{Tcls + 1}$$

maka :

$$\frac{1}{\frac{Ti}{KpK}s - 1} = \frac{1}{Tcls + 1}$$

$$\frac{Ti}{KpK} = Tcl$$

Parameter kontroler dapat diketahui :

$$Ti = \frac{2\,\varsigma}{\omega}$$

$$Td = \frac{1}{Ti\omega^2}$$

$$Kp = \frac{Ti}{TclK}$$

2.1.8 Implementasi Kontroler PID pada Mikrokontroler

Untuk mengimplementasikan kontroler PID ke dalam program mikrokontroler maka fungsi alih kontroler PID harus diruba ke persamaan beda. Fungsi alih PID yaitu :

$$Gc(s) = \frac{C(s)}{E(s)} = Kp\frac{TiTds^2 + Tis + 1}{Tis}$$

diubah dalam domain diskrit z dengan waktu sampling Ts :

$$Gp(z) = \frac{C(z)}{E(z)} = Z\{Gp(s)|_{Ts}\} = \frac{npz}{dpz}$$

$$dp(z)C(z) = np(z)E(z)$$

$$dpz(1)zC(z) + dpz(2)C(z) = npz(1)zE(z) + npz(2)E(z)$$

$$dpz(1)C(z) + dpz(2)z^{-1}y(z) = npz(1)E(z) + npz(2)z^{-1}E(z)$$

$$dpz(1)C(z) = -dpz(2)z^{-1}C(z) + npz(1)E(z) + npz(2)z^{-1}E(z)$$

$$dpz(1)C(n) = -dpz(2)C(n-1) + npz(1)E(n) + npz(2)E(n-1)dpz(1) \times new_{Cn}$$

$$= -dpz(2) \times last_{Cn} + npz(1) \times new_{En} + npz(2) \times last_{En}$$

Persamaan beda Plant:

$$Gp(s) = \frac{K}{\tau pS + 1} = \frac{Y(S)}{C(S)}$$

$$Gp(z) = Z\{Gp(s)|Ts\} = \frac{npz}{dpz} = \frac{Y(s)}{C(s)}$$



$$dp(z)y(z) = np(z)c(z)$$

$$dpz(1)zy(z) + dpz(2)y(z) = npz(1)zc(z) + npz(2)c(z)$$

$$dpz(1)y(z) + dpz(2)z^{-1}y(z) = npz(1)c(z) + npz(2)z^{-1}c(z)$$

$$dpz(1)y(z) = -dpz(2)z^{-1}y(z) + npz(1)c(z) + npz(2)z^{-1}c(z)$$

$$dpz(1)y(n) = -dpz(2)y(n-1) + npz(1)c(n) + npz(2)c(n-1)dpz(1) \times new_{yn}$$

$$= -dpz(2) \times last_{yn} + npz(1) \times new_{cn} + npz(2) \times last_{cn}$$

2.2 Permodelan Sistem Kelistrikan

Arus dan tegangan adalah variabel utama yang digunakan untuk menjelaskan prilaku rangkaian. Arus adalah aliran elektron, yaitu waktu rata-rata muatan elektron yang melewati sebuah daerah tertentu, seperti penampang kawat. Karena elektron muatan yang bersifat negatip, arah positip aliran arus berlawanan dengan alirah elektron. Penjelasan matematika dari hubungan antara sejumlah elektron (muatan q) dan arus (i) adalah :

$$i = \frac{dq}{dt}$$

$$d(q) = \int idt$$

Satuan dari muatan adalah coulomb (C). Satuan arus adalah amper (A), yaitu :

$$arus = \frac{coulomb}{detik}$$

Energi diperlukan untuk menggerakkan sebuah muatan antara dua titik dalam sebuah rangkaian listrik. Kerja per satuan muatan yang dibutuhkan untuk melakukan ini disebut tegangan. Beda tegangan antara dua titik dalam rangkaian adalah sebuah pengukurannergi yang dipakai untuk menggerakkan muatan dari satu ke titik yang lain. Satuan tegangan adalah volt (V), yaitu :

$$tegangan = \frac{joule}{coulomb}$$



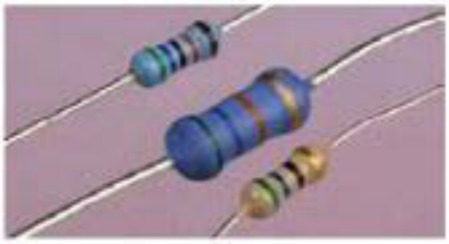
2.2.1 Permodelan Elemen Resistor

Resistansi R dari resistor diberikan oleh :

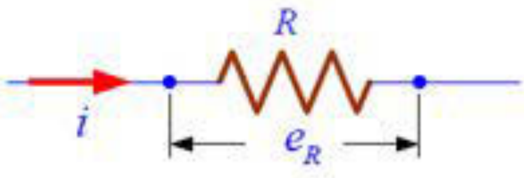
$$R = \frac{e_R}{i}$$

Dimana e_R adalah tegangan pada tahanan dan i adalah arus yang melewati resistor. Satuan resistor adalah ohm (Ω), dimana :

$$ohm = \frac{volt}{amper}$$



(a)



(b)

Gambar 2.8(a) Elemen dan (b) Rangkaian Resistor

2.2.1.1 Persamaan Sistem Resistor

Arus yang mengalir melalui resistor :

$$i = \frac{e_R}{R}$$



2.2.1.2 Fungsi Alih Resistor

Fungsi alih elemen resistor diberikan oleh :

$$I_R(s) = \frac{1}{R} E_R(s)$$

$$\frac{I_R(s)}{E_R(s)} = \frac{1}{R}$$

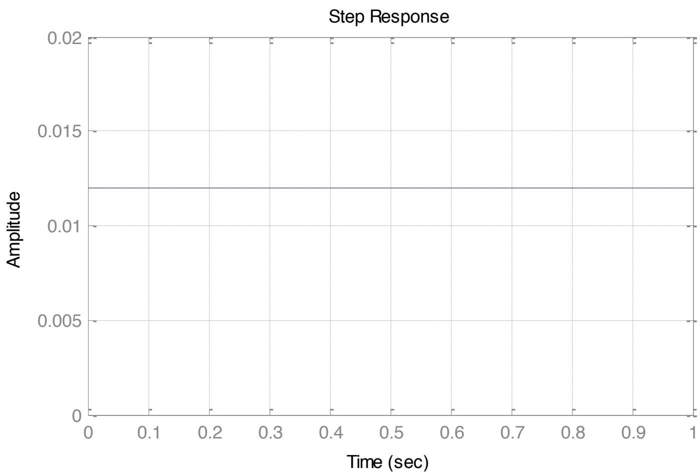
Contoh :

$$I_R(s) = R E_R(s)$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
R=1000;
E=12;
num=[1];
den=[R];
tf_R=tf(num,den);
step(tf_R*E);
grid
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Resistor



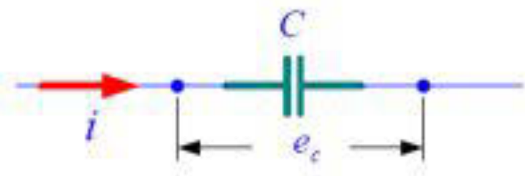
2.2.2 Permodelan Elemen Kapasitor

Dua penghantar dipisahkan oleh medium non penghantar membentuk sebuah kapasitor, jadi dua lempengan metal yang dipisahkan oleh sebuah matrieal dialetrik sangat tipis membentuk sebuah kapasitor.Kapasitansi C adalah sebuah pengukuran besar muatan yang dapat disimpan untuk tegangan yang diberikan terhadap lempengan-lempengan. Kapasitansi C dari sebuah kapasitor dapat diberikan oleh :

$$C = \frac{q}{e_C}$$



(a)



(b)

Gambar 2.8(a) Elemen dan (b) Rangkaian Kapasitor

Dimana q adalah jumlah muatan yang tersimpan dan e_c adalah tegangan pada kapasitor. Satuan kapasitansi adalah Farad (F), dimana :

$$Farad = \frac{coulomb}{volt}$$

2.2.2.1 Persamaan Sistem Kapasitor

Karena diketahui bahwa :

$$i = \frac{dq}{dt}$$

$$e_C = \frac{q}{C}$$



Maka Arus yang mengalir melalui kapasitor :

$$i = C\frac{de_C}{dt}$$

Dan tegangan pada kapasitor :

$$de_C = \frac{1}{C}\int idt + e_C(0)$$

2.2.2.2 Fungsi Alih Kapasitor

Fungsi alih elemen kapasitor diberikan oleh :

$$E_C(s) = \frac{1}{C}\frac{1}{s}I_C(s)$$

$$\frac{E_C(s)}{I_C(s)} = \frac{1}{Cs}$$

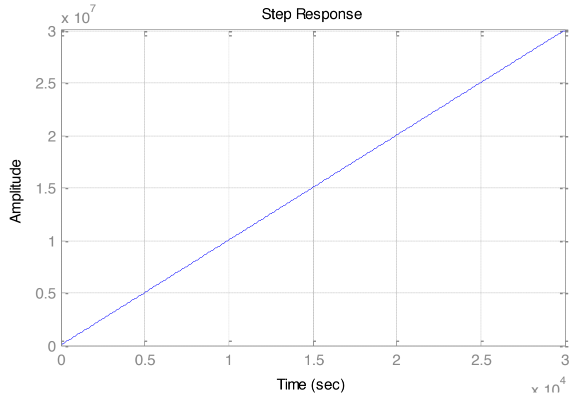
Contoh :

$$E_C(s) = \frac{1}{Cs}I_C(s)$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
C=1*10^-6;
I=1*10^-3;
num=[1];
den=[C 0];
tf_C=tf(num,den)
step(tf_C*I);
grid
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Kapasitor

2.2.3 Permodelan Elemen Induktor

Jika sebuah rangkaian diletakkan dalam sebuah medan magnet yang berubah terhadap waktu, sebuah gaya elektromotif akan diinduksikan dalam rangkaian. Efek induktif dapat diklasifikasikan sebagai induktasi diri dan induktansi mutual. Induktansi diri atau sederhananya induktansi L adalah konstanta proporsional antara tegangan Le volt dan rata-rata pengisian arus per detik dq/dt amper, yaitu

$$L = \frac{e_L}{di/dt}$$

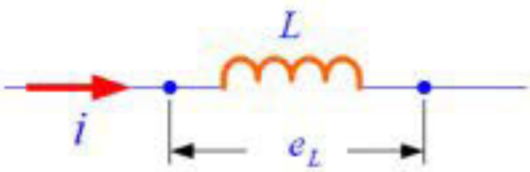
Satuan induktansi adalah henry (H). Sebuah rangkaian listrik mempunyai sebuah induktansi 1 henry jika sebuah rata-rata perubahan 1 amper per detik akan menginduksi sebuah gaya elektromotif (emf) 1 volt.

$$henry = \frac{volt}{amper/detik}$$





(a)



(b)

Gambar 2.8(a) Elemen dan (b) Rangkaian Induktor

2.2.3.1 Persamaan Sistem Induktor

Tegangan e_L pada induktor L diberikan oleh :

$$e_L = L\frac{di_L}{dt}$$

Dimana i_L adalah arus yang melalui induktor. Arus $i_L(t)$ dapat diberikan :

$$i_L(t) = \frac{1}{L}\int e_L\, dt + i_L(0)$$

2.2.3.2 Fungsi Alih Induktor

Fungsi alih elemen induktor diberikan oleh :

$$I_L(s) = \frac{1}{L}\frac{1}{s}E_L(s)$$

$$\frac{E_L(s)}{I_L(s)} = Ls$$

$$\frac{I_L(s)}{E_L(s)} = \frac{1}{Ls}$$

Contoh :

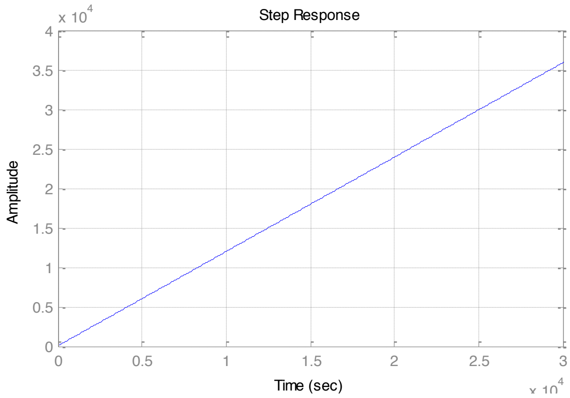
$$I_L(s) = \frac{1}{Ls}E_L(s)$$



Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
L=10;
E=12;
num=[1];
den=[L 0];
tf_L=tf(num,den)
step(tf_L*E);
grid
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

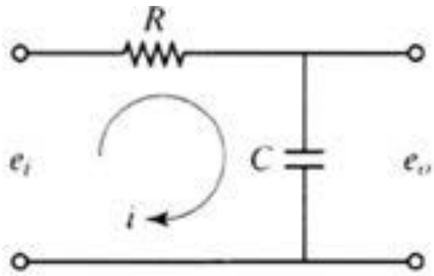


Gambar 2.8Respon Fungsi Alih Induktor



2.2.4 Permodelan Elemen Resistor dan Kapasitor

Rangkaian terdiri dari resistor R dan kapasitor C yang terhubung seri, dengan ei adalah tegangan input dan eo tegangan output.



Gambar 2.8Rangkaian Seri R dan C

2.2.4.1 Persamaan Sistem Resistor dan Kapasitor

Tegangan e_i pada Resistor R dan kapasitor C diberikan oleh :

$$Ri + \frac{1}{C}\int i\,dt = e_i$$

$$\frac{1}{C}\int i\,dt = e_o$$

2.2.4.2 Fungsi Alih Resistor dan Kapasitor

Fungsi alih elemen resistor dan kapasitor diberikan oleh :

$$RI(s) + \frac{1}{Cs}I(s) = E_i(s)$$

$$\frac{1}{Cs}I(s) = E_o(s)$$

$$\frac{E_o(s)}{E_i(s)} = \frac{\frac{1}{Cs}I(s)}{RI(s) + \frac{1}{Cs}I(s)}$$

$$\frac{E_o(s)}{E_i(s)} = \frac{\frac{1}{Cs}}{R + \frac{1}{Cs}} = \frac{1}{RCs + 1}$$

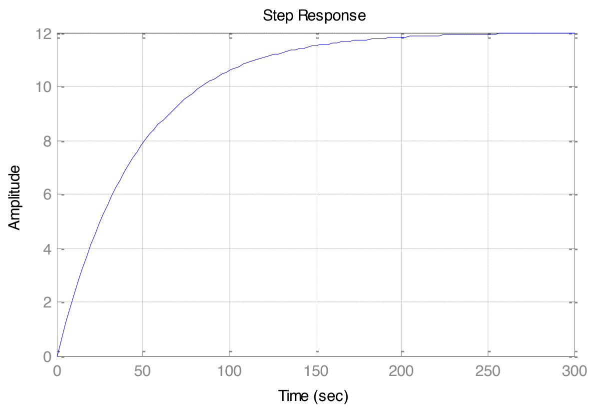
Contoh :

$$E_o(s) = \frac{1}{RCs + 1}E_i(s)$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
R=47000;
C=1000*10^-6;
Eo=12;
num=1;
den=[R*C 1];
tf_RC=tf(num,den);
step(tf_RC*Eo);
grid
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

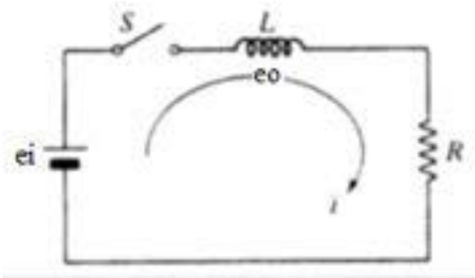


Gambar 2.8Respon Fungsi Alih Resistor dan Kapasitor



2.2.5 Permodelan Elemen Resistor dan Induktor

Rangkaian terdiri dari resistor R dan induktor L yang terhubung seri, dengan e_i adalah tegangan input dan e_o tegangan output.



Gambar 2.8Rangkaian Seri R dan L

2.2.5.1 Persamaan Sistem Resistor dan Kapasitor

Tegangan e_i pada Resistor R dan kapasitor C diberikan oleh :

$$L\frac{di}{dt} + Ri = e_i$$

$$i(t) = \frac{E}{R}\left[1 - e^{-[R/L]t}\right]$$

2.2.5.2 Fungsi Alih Resistor dan Kapasitor

Fungsi alih elemen induktor diberikan oleh :

$$LsI(s) + RI(s) = [Ls + R]I(s) = E_i(s)$$

$$E_i(s) = [Ls + R]I(s)$$

$$\frac{I(s)}{E_i(s)} = \frac{1}{Ls + R}$$

Contoh :

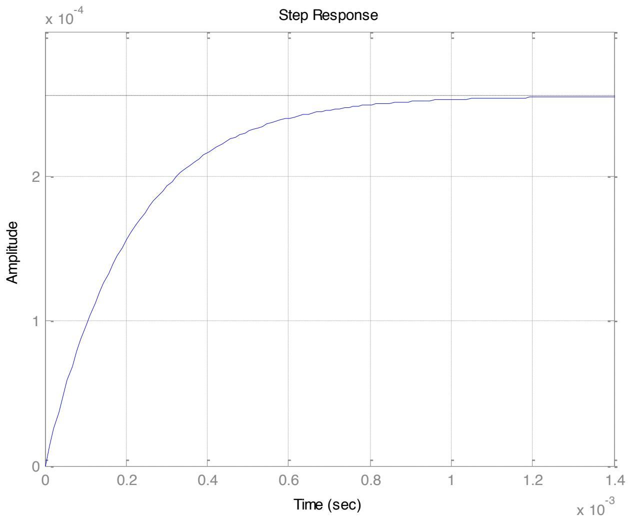
$$I(s) = \frac{1}{Ls + R}E_i(s) = \frac{\frac{1}{R}}{\frac{L}{R}s + 1}E_i$$



Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
R=47000;
C=10;
Eo=12;
num=[1/R];
den=[L/R 1];
tf_RL=tf(num,den);
step(tf_RL*Eo);
grid
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Resistor dan Induktor



Rangkuman

- Persamaan aljabar dasar dapat dipecahkan secara mudah dengan menggunakan aplikasi MATLAB. Berbagai contoh persamaan aljabar seperti persamaan garis lurus, persamaan akar kuadrat, persamaan orde satu, persamaan orde tinggi dan lain-lain dapat mudah dipecahkan dengan menggunakan perintah “solve” dan “roots”.

- Fungsi gambar pada MATLAB berguna untuk merepresentasikan hasil dari suatu persamaan, untuk menggambar hasil suatu persamaan diperlukan langkah-langkah berikut ini :
    - Menentukan x, oleh kisaran tertentu dari nilai x dimana fungsi akan digambarkan
    - Menentukan y = f(x)
    - Memanggil plot untuk gambar fungsi f(x).

- Syntaks perintah gambar pada MATLABterdiri dari beberapa perintah seperti berikut :
    - Menggambar persamaan dua dimensi : Plot (x,y)
    - Menggambar chart bar : bar (x,y)
    - Menggambarkontur :Contour (x,y,g) dimana g=f(x)
    - Menggambar tiga dimensi : surf (x,y,g) dimana g=f(x)



Tugas

Tes Formatif



KEGIATAN 2

2.4 Permodelan Sistem Mekanik

Beberapa system mekanik terdiri dari elemen-elemen mekanik. Tiga jenis elemen-elemen dasar system mekanik adalah :

- Elemen inersia
- Elemen pegas
- Elemen redaman

Sebuah pengertian untuk menyimpan energi kinetik adalah massa atau inersia. Sebuah pengertian untuk menyimpan energi potensial adalah pegas atau elastisitas. Sebuah pengertian yang mana sisipasi energi secara gradual adalah redaman.

Variabel-variabel yang menjadi pengamatan kita adalah :

x : pergeseran (m)

v : kecepatan (m/sec)

a : percepatan (m/sec)

f : gaya (N)

p : kekuatan (Nm/sec)

w : kerja (energi) (Nm)

Semua variabel ini adalah fungsi waktu.

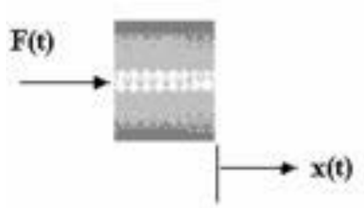
2.4.5 Permodelan Elemen Inersia

Massa dan momen dari elemen inersia Inersia mungkin ditentukan sebagai perubahan gaya (torsi) yang diperlukan untuk membuat sebuah satuan perubahan dalam percepatan (percepatan sudut). Itu adalah :

$$inersia(massa) = \frac{perubahan\ gaya}{perubahan\ percepatan} \frac{N}{m/s^2}\ atau\ kg$$

$$inersia(momen\ inersia) = \frac{perubahan\ torsi}{perubahan\ percepatan\ sudut} \frac{Nm}{rad/s^2}\ atau\ kg$$





Gambar 2.9Rangkaian massa

2.4.5.1 Persamaan ElemenMassa

Gaya pada massa:

$$F = m\,\frac{d^2x}{d^2t}$$

2.4.5.2 Fungsi Alih Massa

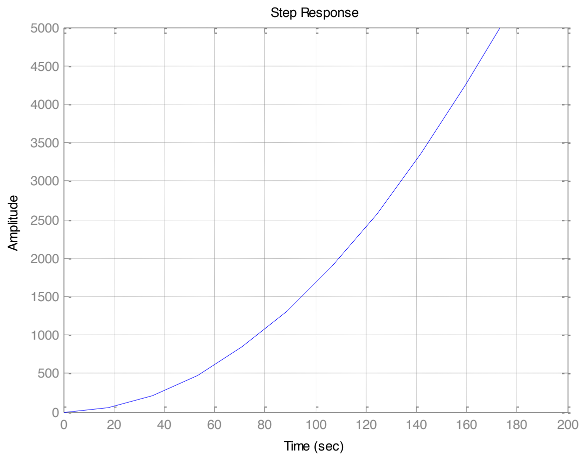
Fungsi alih elemen massa diberikan oleh :

$$\frac{X(s)}{F(s)} = \frac{1}{ms^2}$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
M=3;
F=1;
num=[1];
den=[M 0 0];
tf_M=tf(num,den);
step(tf_M*F);
grid
axis([0 200 0 5000])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Massa

2.4.6 Permodelan Elemen Pegas

Sebuah pegas adalah resistansi dari sebuah benda terhadap defleksi atau deformasi akibat gaya yang diberikan pada benda tersebut. Elemen ini biasanya dikenal sebagai pegas. Gaya kekakuan adalah proporsional terhadap defleksi yang terjadi.Sebuah pegas linier adalah sebuah elemen mekanis yang dapat dibentuk oleh gaya atau torsi dari luar, dimana deformasi adalah proporsional secara langsung terhadap gaya atau torsi yang diberikan pada elemen ini.

2.4.6.1 Persamaan Elemen Pegas Translasi

Untuk gerakan translasi, gaya yang muncul dalam pegas adalah proporsional ke x dan diberikan oleh :

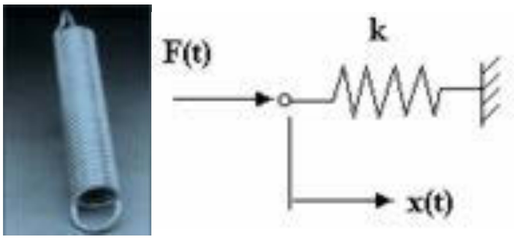
$$F = k\,x$$

Dimana x adalah pemanjangan pegas dan k adalah kontanta proporsional yang disebut konstanta pegas dan satuannya [force/displacement]=[N/m].

Untuk pegas translasi, konstanta pegas k adalah :

$$konstanta\ pegas\ k = \frac{perubahan\ gaya}{perubahan\ pergeseran\ pegas}\frac{N}{m}$$





(a) (b)

Gambar 2.10(a) Elemen dan (b) Rangkaian Pegas

2.4.6.2 Fungsi Alih Pegas Translasi

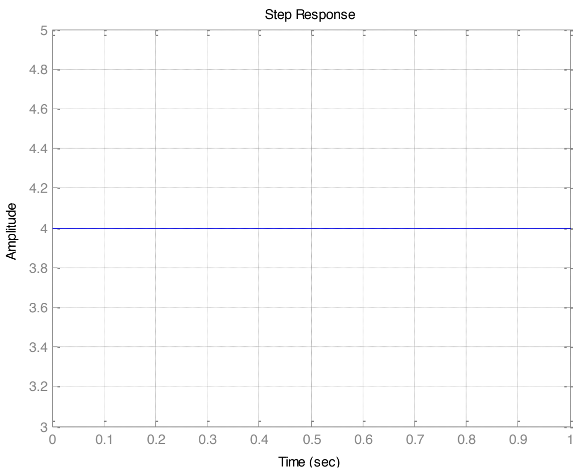
Fungsi alih elemen pegastranslasi diberikan oleh :

$$\frac{X(s)}{F(s)} = \frac{1}{k}$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
k=0.25;
F=1;
num=[1];
den=[k];
tf_k=tf(num,den);
step(tf_k*F);
grid
axis([0 1 0 0.1])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Pegas



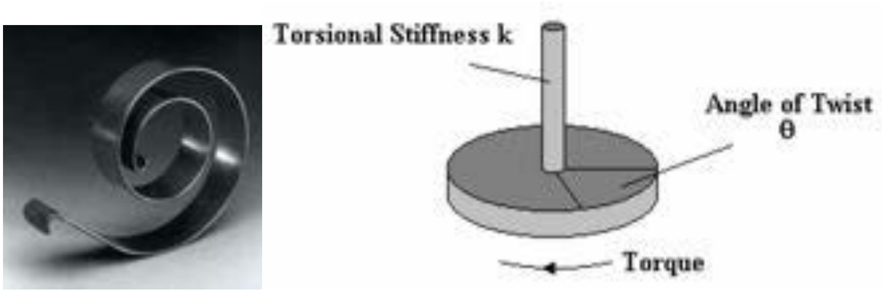
2.4.6.3 Persamaan Elemen Pegas Torsional

Perhatikan pegas torsional, dimana satu ujung adalah tetap dan sebuah torsi τ diberikan pada ujung yang lain. Pergeseran sudut dari ujung yang bebas adalah θ. Torsi T dalam pegas torsional adalah :

$$T = k\,\theta$$

dimana θ adalah pergeseran sudut dan T adalah konstanta pegas untuk pegas torsional dan mempunyai satuan [Torque/angular displacement]=[N-m/rad]. Untuk pegas torsional, konstanta pegas k adalah :

$$konstanta\ pegas\ k = \frac{perubahan\ torsi}{perubahan\ sudut\ pegas}\frac{Nm}{rad}$$



(a) (b)

Gambar 2.10(a) Elemen dan (b) Rangkaian Pegas Torsional

2.4.6.4 Fungsi Alih Pegas Torsional

Fungsi alih elemen pegas torsional diberikan oleh :

$$\frac{\theta\,(s)}{T(s)} = \frac{1}{k}$$

2.4.7 Permodelan Elemen Redaman

Sebuah redaman adalah sebuah elemen mekanis yang mendisipasi energinya dalam pembentukan panas yang tersimpan didalamnya.

2.4.7.1 Persamaan Elemen Redaman Translasi

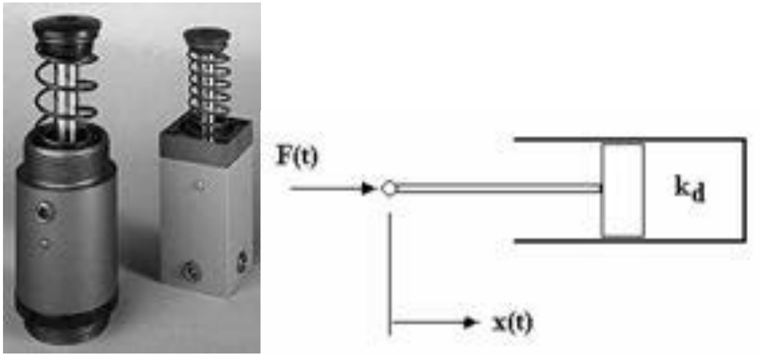
Diagram skematik dari redaman translasi, terdiri dari sebuah piston dan silider berisi oli. Sebuah gerakan relatif antara kepala piston dan silinder dihambat oleh oli, sebab oli harus mengalir disekitar piston dari satu sisi ke sisi yang lain.



Gaya yang diberikan pada ujung dari redaman translasi adalah terjadi pada garis yang sama dan besarnya sama, tetapi arahnya berlawanan. Kecepatan dari ujung redaman adalah $\dot{x}_1\ atau\ \left(\frac{dx1}{dt}\right)$ dan $\dot{x}_2\ atau \left(\frac{dx2}{dt}\right)$. Kecepatan $\dot{x}_1$ dan $\dot{x}_2$ diberikan relatip sama dengan referensi. Dalam redaman translasi, gaya redaman F yang muncul dalamnya adalah proporsional terhadap perbedaan kecepatan $\dot{x}_1$ - $\dot{x}_2$ pada ujung-ujungnya, atau :

$$F = k_d(\dot{x}_1 - \dot{x}_2) = k_d\dot{x}$$

Dimana $(\dot{x}_1 - \dot{x}_2)$dan konstanta redaman b berelasi dengan F pada perbedaan kecepatan adalah $\dot{x}$ disebut koefisien gesekan visko dan satuannya [force/kecepatan]=[Ns/m].



(a) (b)

Gambar 2.10(a) Elemen dan (b) Rangkaian Redaman Translasi

2.4.7.2 Fungsi Alih RedamanTranslasi

Fungsi alih elemen redamantranslasi diberikan oleh :

$$F(s) = k_d sX(s)$$

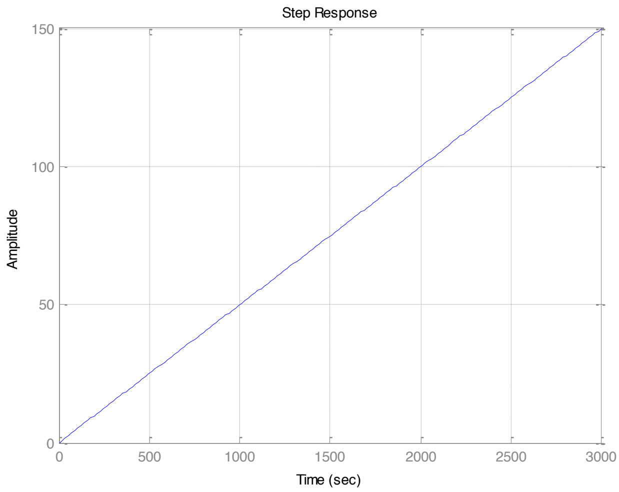
$$\frac{X(s)}{F(s)} = \frac{1}{k_d s}$$



Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
kd=20;
F=1;
num=[1];
den=[kd 0];
tf_g=tf(num,den);
step(tf_g*F);
grid
axis([0 3000.2 0 150])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Redaman

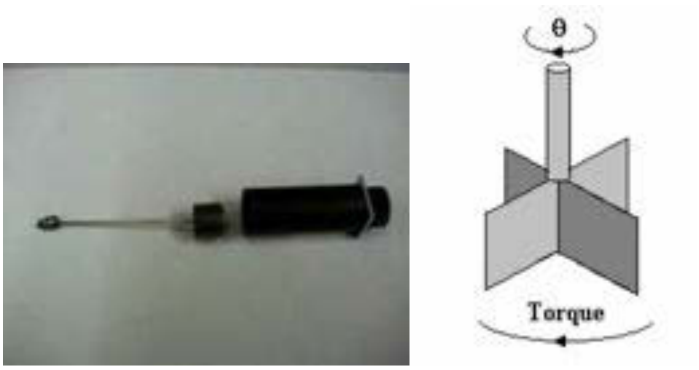


2.4.7.3 Persamaan Elemen Redaman Torsional

Pada redaman torsional, torsi τ yang diberikan pada ujung dari redaman translasi adalah terjadi pada garis yang sama dan besarnya sama, tetapi arahnya berlawanan. Kecepatan sudut dari ujung redaman torsional adalah $\dot{\theta}_1\ atau\ \left(\frac{d\theta 1}{dt}\right)$ dan $\dot{\theta}_2\ atau \left(\frac{d\theta 2}{dt}\right)$. Kecepatan $\dot{\theta}_1$ dan $\dot{\theta}_2$ diberikan relatip sama dengan referensi. Dalam redaman torsional, torsi redaman τ yang muncul dalamnya adalah proporsional terhadap perbedaan kecepatan $\dot{\theta}_1$ - $\dot{\theta}_2$ pada ujung-ujungnya, atau :

$$F = k_d(\dot{\theta}_1 - \dot{\theta}_2) = k_d\dot{\theta}$$

Dimana $(\dot{\theta}_1 - \dot{\theta}_2)$dan konstanta redaman b berelasi dengan τ pada perbedaan kecepatan sudut adalah $\dot{\theta}$ disebut koefisien gesekan visko dan satuannya [torsi/kecepatan sudut]=[Nms/rad].



(a) (b)

Gambar 2.10(a) Elemen dan (b) Rangkaian Redaman Torsional



2.4.7.4 Fungsi Alih Redaman Torsional

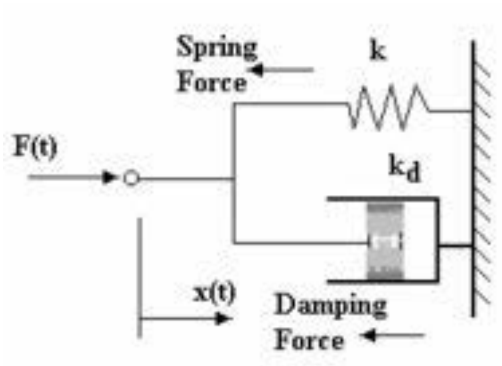
Fungsi alih elemen redaman torsional diberikan oleh :

$$\tau(s) = k_d s\theta(s)$$

$$\frac{\theta(s)}{\Gamma(s)} = \frac{1}{k_d s}$$

2.4.8 Permodelan Elemen Pegas dan Redaman

Rangkaian terdiri dari pegas dan redaman yang terhubung paralel, dengan f adalah gaya input dan x jarak output.



Gambar 2.8Rangkaian paralel Pegas dan Redaman

2.4.8.1 Persamaan Sistem Pegas dan Redaman

Tegangan f pada pegas dan redaman diberikan oleh :

$$f(t) = kx + k_d \frac{dx}{dt}$$

2.4.8.2 Fungsi Alih Pegas dan Redaman

Fungsi alih elemen pegas dan redaman diberikan oleh :

$$F(s) = kX(s) + k_d sX(s)$$

$$\frac{X(s)}{F(s)} = \frac{\frac{1}{k}}{\frac{kd}{k}s + 1}$$



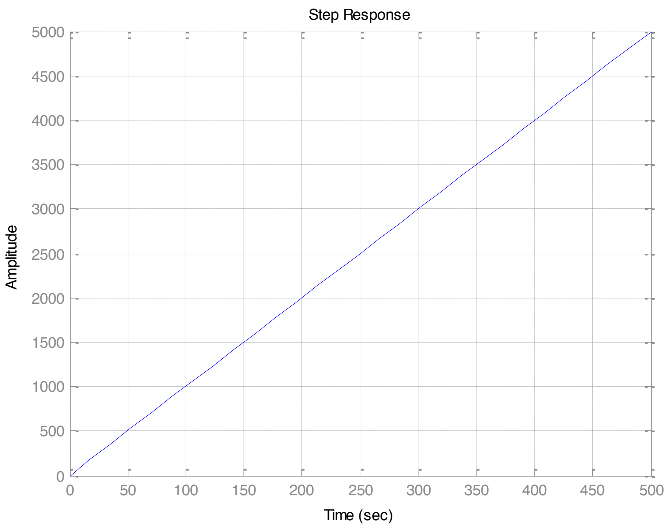
Contoh :

$$X(s) = \frac{\frac{1}{k}}{\frac{kd}{k}s + 1}F(s)$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
k=25;
d=0.10
F=1;
num=[1/k];
den=[d/k 0];
tf_kd=tf(num,den);
step(tf_kd*F);
grid
axis([0 500 0 5000])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

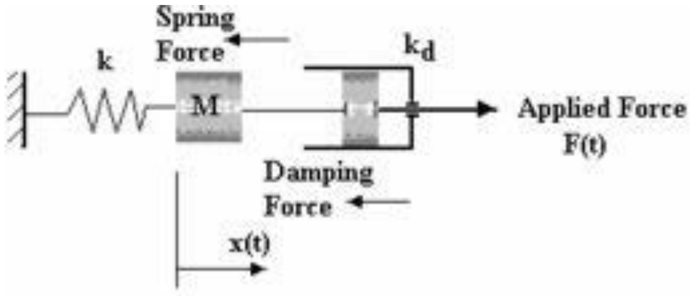


Gambar 2.8Respon Fungsi Alih Pegas dan Redaman



2.4.9 Permodelan Elemen Massa, Pegas dan Redaman

Rangkaian terdiri dari massa, pegas dan redaman yang terhubung seri , dengan f adalah gaya input dan x jarak output.



Gambar 2.8Rangkaian paralel Massa, Pegas dan Redaman

2.4.9.1 Persamaan Sistem Massa, Pegas dan Redaman

Tegangan f pada Massa, pegas dan redaman diberikan oleh :

$$f(t) = M\frac{d^2x}{d^2t} + k_d\frac{dx}{dt} + kx$$

2.4.9.2 Fungsi Alih Massa, Pegas dan Redaman

Fungsi alih elemen massa, pegas dan redaman diberikan oleh :

$$F(s) = Ms^2X(s) + k_d sX(s) + kX(s)$$

$$\frac{X(s)}{F(s)} = \frac{\frac{1}{k}}{\frac{M}{k}s^2 + \frac{kd}{k}s + 1}$$



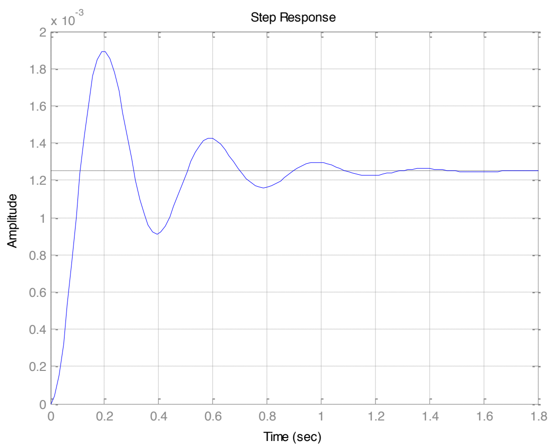
Contoh :

$$X(s) = \frac{\frac{1}{k}}{\frac{M}{k}s^2 + \frac{kd}{k}s + 1}F(s)$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
F=1;
M=3;
k=800;
kd=20;
mpg=40;
num=[1/k];
den=[M/k kd/k 1];
tf_mpg=tf(num,den);
step(tf_mpg*F);
grid
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Massa, Pegas dan Redaman



Rangkuman

- Persamaan aljabar dasar dapat dipecahkan secara mudah dengan menggunakan aplikasi MATLAB. Berbagai contoh persamaan aljabar seperti persamaan garis lurus, persamaan akar kuadrat, persamaan orde satu, persamaan orde tinggi dan lain-lain dapat mudah dipecahkan dengan menggunakan perintah “solve” dan “roots”.

- Fungsi gambar pada MATLAB berguna untuk merepresentasikan hasil dari suatu persamaan, untuk menggambar hasil suatu persamaan diperlukan langkah-langkah berikut ini :
  - Menentukan x, oleh kisaran tertentu dari nilai x dimana fungsi akan digambarkan
  - Menentukan y = f(x)
  - Memanggil plot untuk gambar fungsi f(x).

- Syntaks perintah gambar pada MATLABterdiri dari beberapa perintah seperti berikut :
  - Menggambar persamaan dua dimensi : Plot (x,y)
  - Menggambar chart bar : bar (x,y)
  - Menggambarkontur :Contour (x,y,g) dimana g=f(x)
  - Menggambar tiga dimensi : surf (x,y,g) dimana g=f(x)



Tugas

Tes Formatif



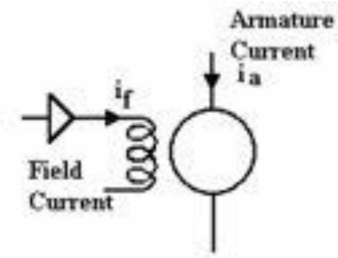
KEGIATAN 3

2.5 Permodelan Sistem Motor DC

Sebuah aktuator sistem motor DC adalah memberikan gerakan putar dan posisi.

2.5.1 Permodelan Sistem Motor DC Kontrol Medan

Sebuah aktuator sistem motor DC dengan kontrol medan magnet adalah :



Gambar 2.8Rangkaian Kelistrikan Motor DC dengan Medan Terkontrol

2.5.1.1 Persamaan Sistem Motor DC Kontrol Medan

Torsi putar T pada sistem motor DC kontrol medan diberikan oleh :

$$T = k_f i_f$$

Jika motor menggerakkan sebuah beban inersia dan mempunyai gesekan, maka persamaan menjadi :

$$T = I\frac{d^2\theta}{d^2t} + k_d\frac{d\theta}{dt}$$

2.5.1.2 Fungsi Alih Sistem Motor DC Kontrol Medan

Fungsi alih elemen sistem motor DC kontrol medan diberikan oleh :

$$T(s) = k_f I_f(s)$$

$$T(s) = Is^2\theta(s) + k_d s\theta(s)$$

$$T(s) = (Is^2 + k_d s)\theta(s)$$

$$(Is^2 + k_d s)\theta(s) = k_f I_f(s)$$

$$\frac{\theta(s)}{I_f(s)} = \frac{k_f}{Is^2 + k_d s}$$



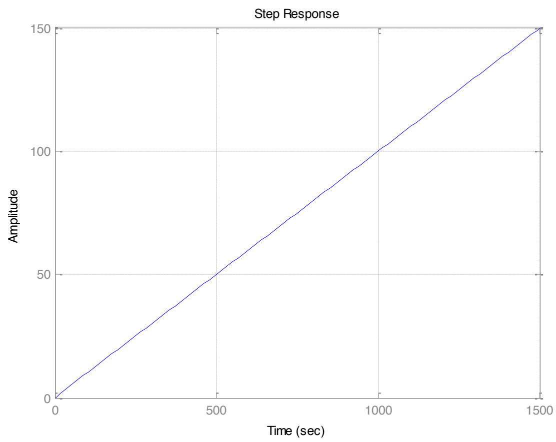
Contoh :

$$\frac{\theta(s)}{I_f(s)} = \frac{k_f}{Is^2 + k_d s}$$

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
I=0.01;
kf=0.01
kd=0.1;
i_f=1;
num=[kf];
den=[I kd 0];
tf_g=tf(num,den);
step(tf_g*i_f);
grid
axis([0 1500.2 0 150])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :

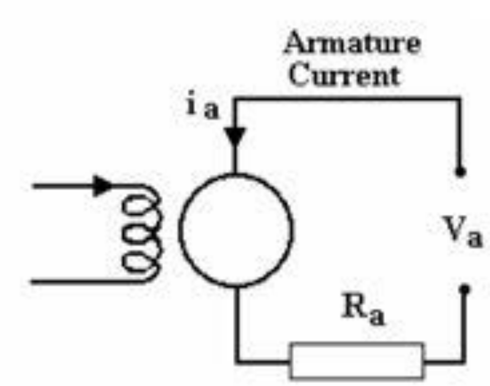


Gambar 2.8Respon Fungsi Alih Sistem Motor DC Kontrol Medan



2.5.2 Permodelan Sistem Motor DC Kontrol Armatur

Sebuah aktuator sistem motor DC dengan kontrol armaturadalah :



Gambar 2.8Rangkaian Kelistrikan Motor DC dengan ArmaturTerkontrol

2.5.2.1 Persamaan Sistem Motor DC Kontrol Armatur

Torsi putar pada motor DC kontrol armaturyaitu relasi tegangan dan resistansi yang ditunjukkan oleh :

$$T = \left(V_a - k\frac{d\theta}{dt}\right)\frac{k}{Ra}$$

Torsi harus menanggulangi kerja inesia dan gesekan :

$$T = I\frac{d^2\theta}{d^2t} + k_d\frac{d\theta}{dt}$$

Sehingga persamaan torsi menjadi :

$$I\frac{d^2\theta}{d^2t} + k_d\frac{d\theta}{dt} = \left(V_a - k\frac{d\theta}{dt}\right)\frac{k}{Ra}$$

2.5.2.2 Fungsi Alih Sistem Motor DC Kontrol Armatur

Fungsi alih elemen motor DC kontrol armatur diberikan oleh :

$$T(s) = (Is^2 + k_d s)\theta(s) = \left(V_a - ks\theta(s)\right)\frac{k}{Ra}$$

$$(Is^2 + k_d s)\theta(s) = V_a\frac{k}{Ra} - \frac{k^2}{Ra}s\theta(s)$$

$$\left(Is^2 + k_d s + \frac{k^2}{Ra}\right)\theta(s) = V_a\frac{k}{Ra}$$



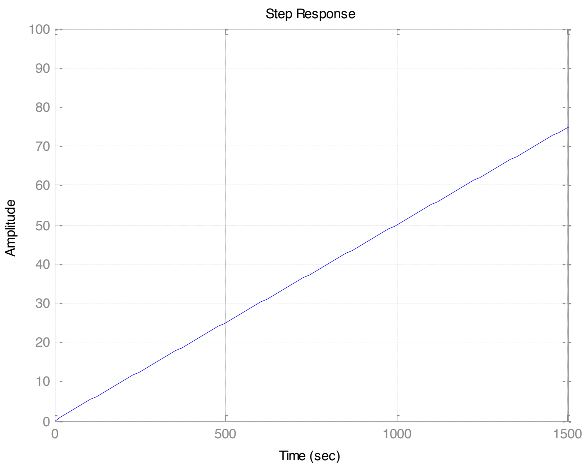
$$\frac{\theta(s)}{V_a(s)} = \frac{\frac{k}{Ra}}{Is^2 + \left(k_d + \frac{k^2}{Ra}\right)s}$$

Contoh :

Buat file script dan ketikan perintah-perintah dibawah ini :

```
clc;
I=0.01;
k=0.01;
kd=0.1;
Ra=2;
Va=1;
num=[k/Ra];
den=[I kd+(k^2/Ra) 0];
tf_a=tf(num,den);
step(tf_a*Va);
grid
axis([0 1500.2 0 100])
```

Setelah dijalankan, jendela perintah akan menampilkan hasil :



Gambar 2.8Respon Fungsi Alih Sistem Motor DC Kontrol Armatur

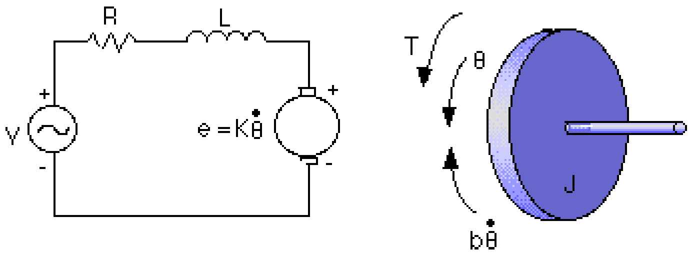


2.5.3 Permodelan Putaran Sistem Motor DC

Sebuah aktuator sistem motor DC adalah memberikan gerakan putar dan dikopel dengan piringan atau belt, dapat memberikan gerakan transional. Rangkaian listrik armatur dan rotor yang bergerak bebas ditunjukkan pada gambar berikut ini :



(a)



(b)

Gambar 2.8(a) Fisik Motor DC Penghasil Putaran, (b) Rangkaian Kelistrikan Motor DC

Untuk contoh ini, kita asumsikan nilai-nilai berikut sebagai parameter fisik motor listrik.



Nilai-nilai ini diturunkan dari hasil percobaan motor sesungguhnya.

- Momen inersia rotor (J) = 0.01 kg.m^2/s^2
- Rasio peredaman sistem mekanik (b) = 0.1 Nms
- Konstanta gaya electromotif (K=Ke=Kt) = 0.01 Nm/Amp
- Tahanan elektrik (R) = 1 ohm
- Induktansi elektrik (L) = 0.5 H
- Input tegangant (V): sumber tegangan
- Output (theta): posisi poros motor
- Rotor dan poros diasumsikan rigid

2.5.3.1 Persamaan SistemPutaran Sistem Motor DC

Torsi motor T, direlasikan dengan arus armatur i, oleh faktor konstatnta Kt.Tegangan balik emf e, direlasikan dengan kecepatan putar oleh persamaan:

$$T = K_t i$$

$$e = K_e \frac{d\theta}{dt}$$

Dalam satuan SI, Kt (konstanta armatur) sama dengan Ke (konstanta motor con).

Dari gambar rangkaian diatas, kita dapat tuliskan persamaan berikut berdasarkan hukum Newton dikombinasikan dengan hukum Kirchhoff:

$$J\frac{d\theta}{dt} + b\frac{d^2\theta}{d^2t} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\frac{d\theta}{dt}$$

2.5.3.2 Fungsi Alih Putaran Sistem Motor DC

Menggunakan transformasi laplace, persamaan permodelan diatas dapat diekspresikan dalam bentuk s .

$$(Js^2 + bs)\theta(s) = KI(s)$$

$$(Ls + R)I(s) = V - Ks\theta(s)$$



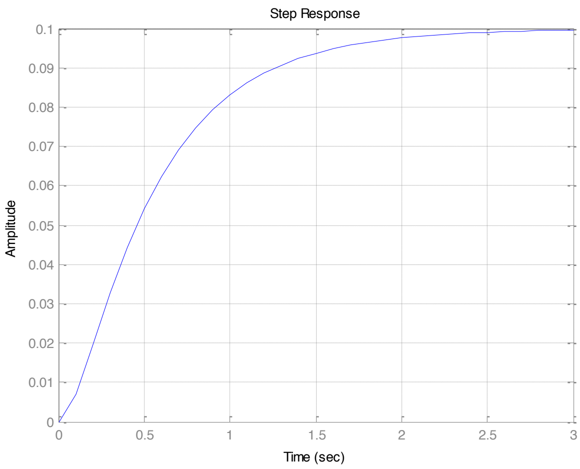
Dengan eliminasi I(s), kita dapatkan fungsi alih rangkaian terbuka, dimana kecepatan putar adalah output dan tegangan adalah input.

$$\frac{\theta(s)}{V(s)} = \frac{K}{(Js+b)(Ls+R)+K^2}$$

Buatlah skrip seperti dibawah ini :

```
clc;clf
J=0.01;
b=0.1;
K=0.01;
R=1;
L=0.5;
num=K;
den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2)];
step(num,den,0:0.1:3);
grid
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 2.9Respon Kecepatan Putar Motor DC

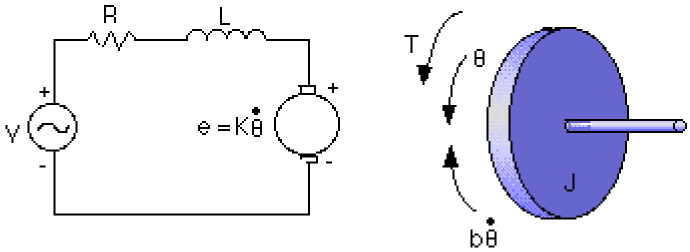


2.5.4 Permodelan Posisi Sistem Motor DC

Sebuah aktuator sistem motor DC adalah memberikan gerakan putar dan dikopel dengan piringan atau belt, dapat memberikan gerakan transional. Rangkaian listrik armatur dan rotor yang bergerak bebas ditunjukkan pada gambar berikut ini :



(a)



(b)

Gambar 2.9(a) Fisik Motor DC, (b) Rangkaian Kelistrikan Motor DC

Untuk contoh ini, kita asumsikan nilai-nilai berikut sebagai parameter fisik motor listrik.

Nilai-nilai ini diturunkan dari hasil percobaan motor sesungguhnya.

- Momen inersia rotor (J) = 3.2284E-6 kg.m^2/s^2
- Rasio peredaman sistem mekanik (b) = = 3.5077E-6 Nms
- Konstanta gaya electromotif (K=Ke=Kt) = 0.0274 Nm/Amp
- Tahanan elektrik (R) = 4 ohm
- Induktansi elektrik (L) = 2.75E-6 H
- Input tegangant (V): sumber tegangan



- Output (theta): posisi poros motor
- Rotor dan poros diasumsikan rigid

2.5.4.1 Persamaan Sistem Posisi Sistem Motor DC

Torsi motor T, direlasikan dengan arus armatur i, oleh faktor konstatnta Kt. Tegangan balik emf e, direlasikan dengan kecepatan putar oleh persamaan:

$$T = K_t i$$

$$e = K_e \frac{d\theta}{dt}$$

Dalam satuan SI, Kt (konstanta armatur) sama dengan Ke (konstanta motor con).

Dari gambar rangkaian diatas, kita dapat tuliskan persamaan berikut berdasarkan hukum Newton dikombinasikan dengan hukum Kirchhoff:

$$J\frac{d\theta}{dt} + b\frac{d^2\theta}{d^2t} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\frac{d\theta}{dt}$$

2.5.4.2 Fungsi AlihPosisi Sistem Motor DC

Menggunakan transformasi laplace, persamaan permodelan diatas dapat diekspresikan dalam bentuk s .

$$(Js^2 + bs)\theta(s) = KI(s)$$

$$(Ls + R)I(s) = V - Ks\theta(s)$$

Dengan eliminasi I(s), kita dapatkan fungsi alih rangkaian terbuka, dimana kecepatan putar adalah output dan tegangan adalah input.

$$\frac{\theta(s)}{V(s)} = \frac{K}{(Js + b)(Ls + R) + K^2}$$

Kita dapat menentukan posisi dengan mengintegrasikan $\theta(s)$, dengan demikian kita membagi fungsi alih dengan s.

$$\frac{\theta(s)}{V(s)} = \frac{K}{s\big((Js + b)(Ls + R) + K^2\big)}$$



Buatlah skrip seperti dibawah ini :

```
J=3.2284E-6;

b=3.5077E-6;

K=0.0274;

R=4;

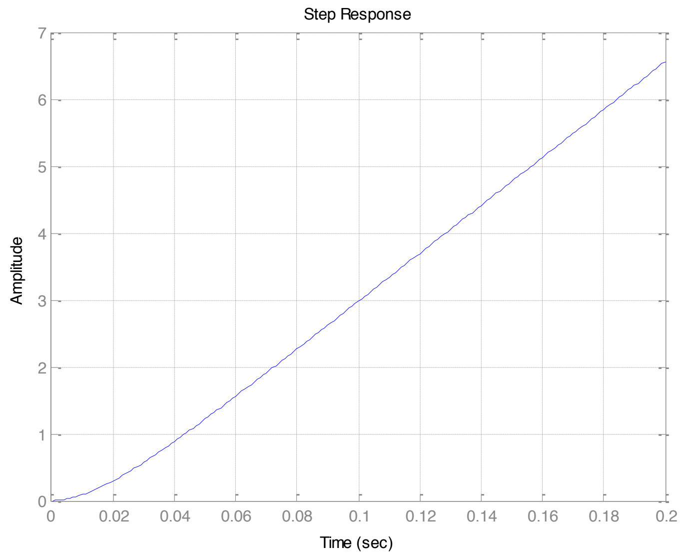
L=2.75E-6;

num=K;

den=[(J*L) ((J*R)+(L*b)) ((b*R)+K^2) 0];

step(num,den,0:0.001:0.2)
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



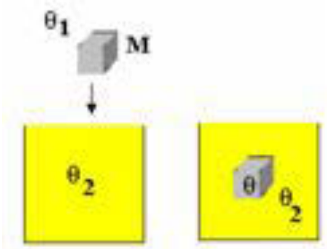
Gambar 2.11Respon Posisi Motor DC

Dari gambar diatas, kita dapat lihat bahwa apabila tegangan 1 volt diberikan ke sistem, maka motor dapat memberikan perubahan posisi motor 6 radian, 6 kali lebih besar dari posisi yang diinginkan



2.5.5 Permodelan Sistem Panas

Amati sebuah massa M kg pada suhuθ_1. Massa diletakkan dalam sebuah lingkungan panas dengan suhu θ_2 dan panas Q ditransfer kepada nassa yang disebabkan suhunya yang naik.



Gambar 2.13Rangkaian Model Transfer Panas

2.5.5.1 Persamaan Sistem Panas

Hukum transfer panas menjelaskan bahwa suhu muncul proporsional secara langsung pada penambahan panas :

$$\frac{dq}{dt} = Mc\frac{\theta_1}{dt} = C\frac{\theta_1}{dt}$$

C adalah kapasitas panas tertentu. C=Mc adalah kapasitansi panas dalam satuan [Joule/Kelvin].

$$\frac{dQ}{dt} = \Phi = C\frac{d\theta_1}{dt}$$

Rata-rata transfer panas ke massa adalah Φ dan rata-rata resistansi yang dibentuk antara cairan dan massa. R adalah resistansi panas dalam satuan [Kelvin/Watt].

$$\Phi = \frac{\theta_2 - \theta_1}{R}$$

$$C\frac{d\theta_1}{dt} = \frac{\theta_2-\theta_1}{R} \text{dan} \frac{d\theta_1}{dt} = \frac{\theta_2-\theta_1}{RC}$$

$$\frac{d\theta_1}{dt} + \frac{\theta_1}{RC} = \frac{\theta_2}{RC}$$

T =RC adalah konstanta waktu

$$\frac{d\theta_1}{dt} + \frac{\theta_1}{T} = \frac{\theta_2}{T}$$



2.5.5.2 Fungsi Alih Sistem Panas

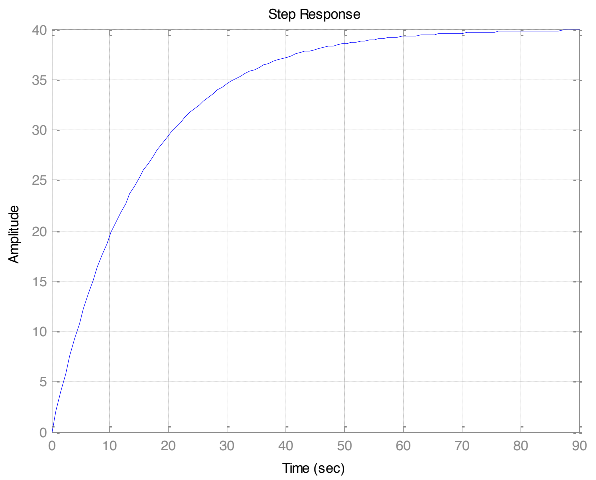
Fungsi alih sistem panas :

$$\frac{\theta_2}{\theta_1} = \frac{1}{Ts + 1}$$

Buatlah skrip seperti dibawah ini :

```
clc;clf
R=0.5;
C=30;
T=R*C;
H1=40;
num=[1];
den=[T 1];
tf_H=tf(num,den);
step(tf_H*H1);
grid
```

Setelah dijalankan, MATLAB akan menampilkan hasil :



Gambar 2.13Rangkaian Model Transfer Panas



Rangkuman

- Persamaan aljabar dasar dapat dipecahkan secara mudah dengan menggunakan aplikasi MATLAB. Berbagai contoh persamaan aljabar seperti persamaan garis lurus, persamaan akar kuadrat, persamaan orde satu, persamaan orde tinggi dan lain-lain dapat mudah dipecahkan dengan menggunakan perintah “solve” dan “roots”.

- Fungsi gambar pada MATLAB berguna untuk merepresentasikan hasil dari suatu persamaan, untuk menggambar hasil suatu persamaan diperlukan langkah-langkah berikut ini :
  - Menentukan x, oleh kisaran tertentu dari nilai x dimana fungsi akan digambarkan
  - Menentukan y = f(x)
  - Memanggil plot untuk gambar fungsi f(x).

- Syntaks perintah gambar pada MATLABterdiri dari beberapa perintah seperti berikut :
  - Menggambar persamaan dua dimensi : Plot (x,y)
  - Menggambar chart bar : bar (x,y)
  - Menggambarkontur :Contour (x,y,g) dimana g=f(x)
  - Menggambar tiga dimensi : surf (x,y,g) dimana g=f(x)



Tugas

Tes Formatif



KEGIATAN 4

2.6 Komponen Kontrol

Pada sistem kontrol, baik sistem kontrol terbuka maupun kontrol tertutup memerlukan beberapa peralatan kontrol yang dapat menghubungkan beberapa sistem mulai dari input sampai pada output.

2.6.1 Baterai

Baterai digunakan dalam kebutuhan elektronik sebagai sumber daya tegangan kimiawi. Sebuah baterai terdiri dari satu atau sel eletro kimiawi yang mana reaksi kimia menghasilkan beda potensial (tegangan) antara terminal-terminalnya. Tegangannya dapat habis terpakai jika arusnya melewati beban yang terpasang pada baterai. Terdapat dua jenis baterai :

- Baterai tetap, biasa disebut sebagai baterai primer dimana mereka tidak bisa diisi kembali tegangannya.
- Baterai dapat diisi kembali, mereka dapat diisi kembali melalui terminal-terminalnya dari sumber tegangan eksternal.

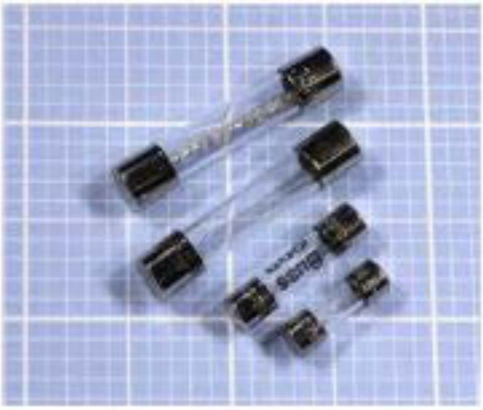


Gambar 2.13 Rangkaian Model Transfer Panas

2.6.2 Sekering

Elemen dalam sekering biasanya adalah sebuah kawat atau logam tipis yang menghubungkan kedua terminalnya, elemen ini terbungkus dalam sebuah silider gelas atau keramik dengan konta-kontak pada kedua ujungnya. Sekering hanya merespon arus besar arus yang mengalir dan bukan tegangannya. Elemen sekering akan terputus apabila arus yang mengalir didalamnya melebihi kemampuan daya hantar arus nominalnya.

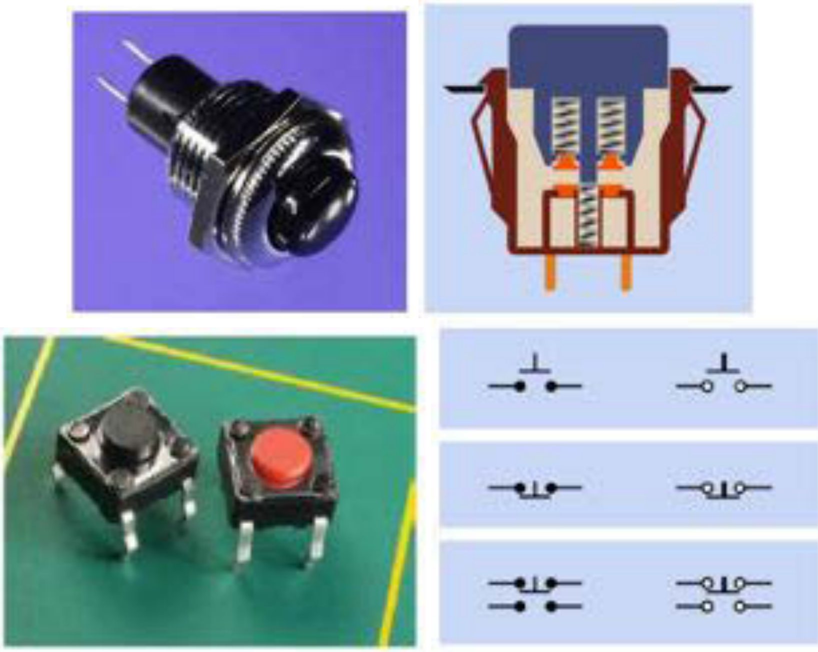




Gambar 2.13 Rangkaian Model Transfer Panas

2.6.3 Tombol

Tombol terdiri sedikitnya dua kontak, dimana terhubung atau terputus bila tombol ditekan Biasanya sebuah pegas mengembalikan tombol pada posisinya apabila tekanan luar dilepas.



Gambar 2.13 Rangkaian Model Transfer Panas

Perilaku On-off tombol terdiri dari beberapa macam :

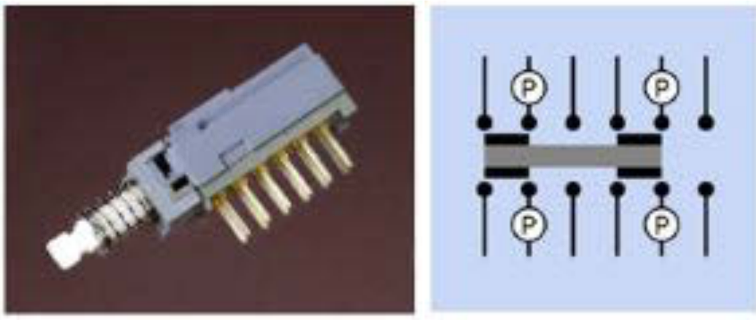
- OFF-ON, kontak secara normal terputus dan terhubung hanya apabila tombol ditekan.
- ON-OFF, kontak secara normal tehubung dan terputus apabila tombol ditekan.



- ON-OFF dan OFF-ON, kontak 1 secara normal terhubung, kontak 2 secara normal terputus dan kontak 1 akan terputus , kontak 2 terhung apabila tombol ditekan.

2.6.4 Tombol Geser

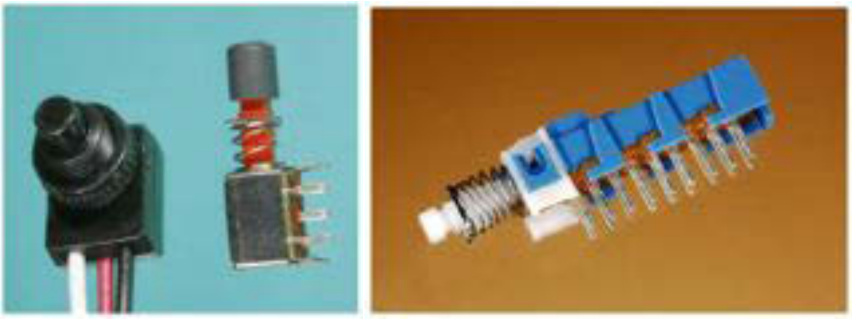
Tombol jenis ini terdiri dari sebuah batang tipis yang digeser masuk dan keluar pada tabung yang panjang dan sempit. Empat kutup tombol tekan ganda, akan menghubungkan dan memutuskan empat pasang kutub bila tombol ditekan atau dilepas.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.5 Tombol Terkunci

Jenis yang lain, juga dikenal dengan tombol dua kali tekan, terdiri sebuah menkanik pengunci, yang mana berputar setiap kali tombol ditekan. Tekanan pertama menyebabkan kontak terkunci pada keadaan tehubung. Tekanan kedua mengembalikan kontah pada keadaan terputus.

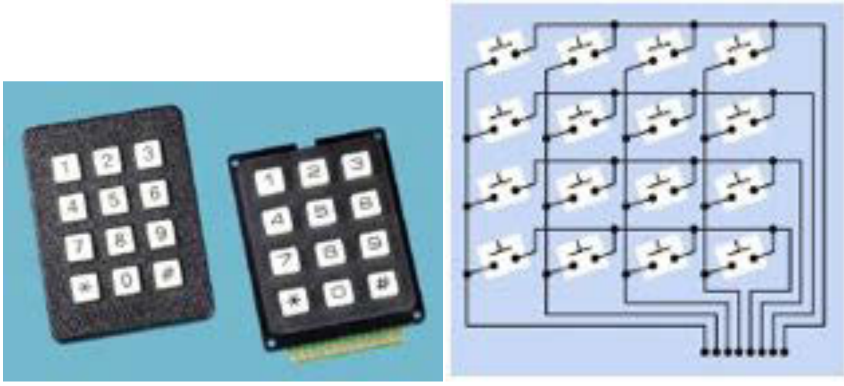


Gambar 2.13 Rangkaian Model Transfer Panas



2.6.6 Keypad

Sebuah keypad adalah sebuah deretan persegi empat dari biasanya 12 atau 16 tombol OFF-ON. Kontak mereka diakses melalui sebuah header yang sesuai untuk terhubung dengan sebuah kabel pita yang tersisip pada papan rangkaian tercetak

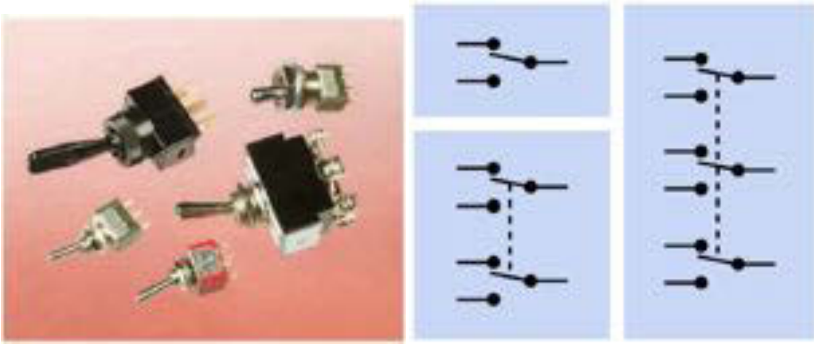


Gambar 2.13 Rangkaian Model Transfer Panas

2.6.7 Sakelar

Kata ON dan OFF digunakan untuk menunjukkan kemungkinan keadaan sebuah sakelar. Tambahan kata NONE digunkana oleh beberapa pabrik untuk menunjukkan bahwa sakelar tidak mempunyai sebuah pusat posisi.

- ON-OFF atau ON-NONE-OFF, sebuah dasar SPST, ON-OFF dengan tidak ada pusat posisi.
- ON-ON atau ON-NONE-ON, sebuah dasar SPDT, dengan tidak mempunyai pusat posisi.
- ON-OFF-ON, sebuah sakelar ganda dengan pusat posisi OFF.

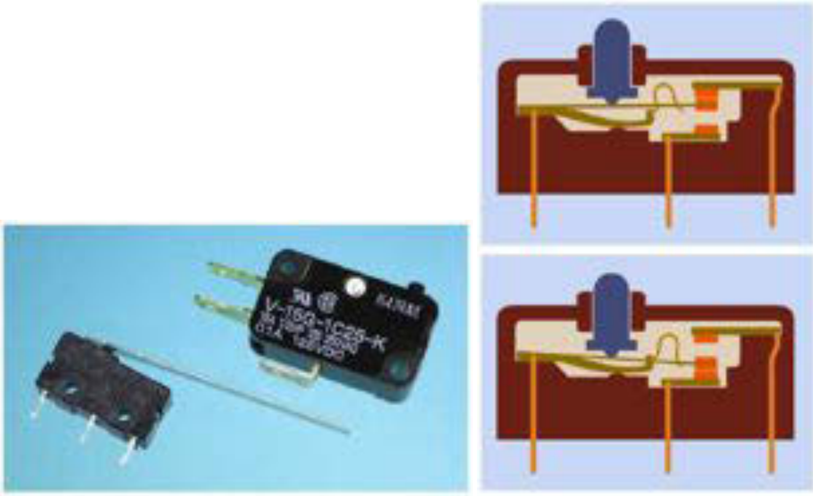


Gambar 2.13 Rangkaian Model Transfer Panas



2.6.8 Limit Switch

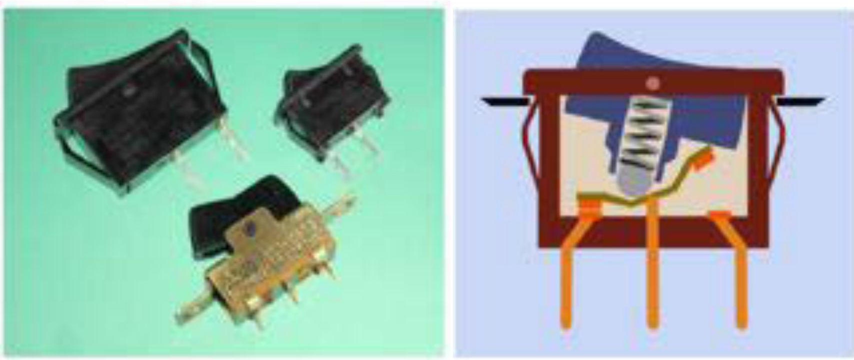
Juga dikenal sebagai sebuah microswitch. Limit switch menggunakan sebuah mekanis pegas pengangkat internal yang mana akan menangkap dua keadaan posisi tertentu. Tipe ini adalah sakelar SPDT biasa dan mempunyai sebuah aksi sesaat.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.9 Sakelar Geser

Banyak jenis sakelar geser yang secara luas digunakan karena harganya murah, tetapi tidak cocok dipakai sebagai komponen kontrol elektronik kecil. Kebanyakan sakelar geser mempunyai dua posisi dan fungsi sebagai SPDT atau DPDT, tetapi konfigurasi lain adalah kurang umum dengan banyak posisi.

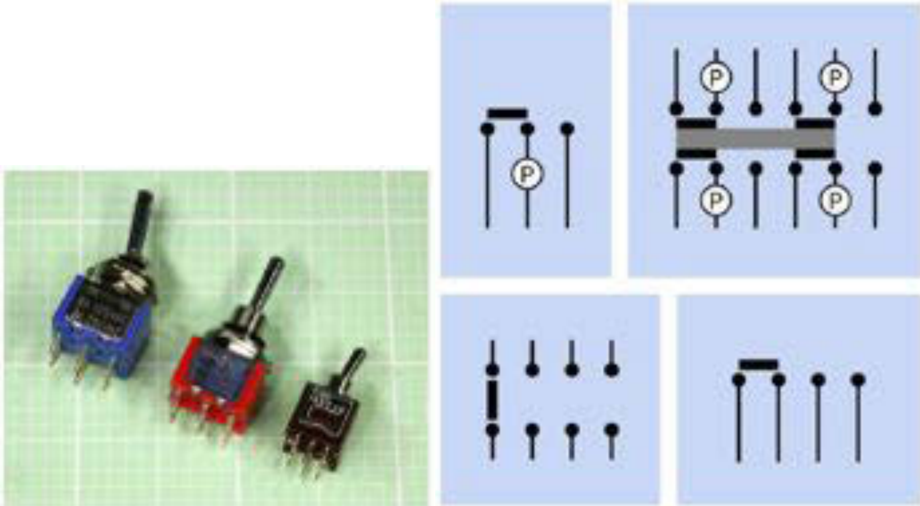


Gambar 2.13 Rangkaian Model Transfer Panas



2.6.10 Sakelar Togel

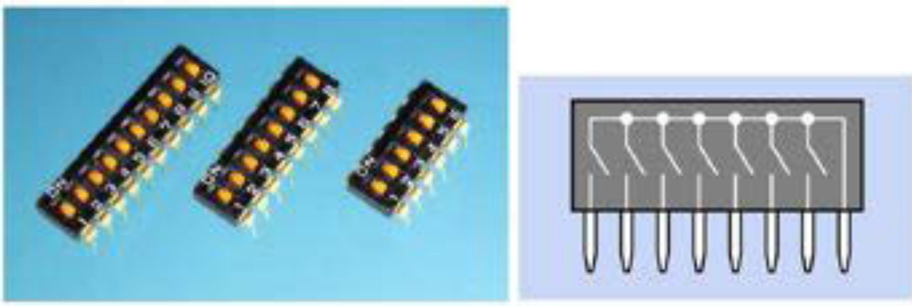
Sakelar togel memberikan sebuah aksi presisi melalui sebuah tongkat yang biasanya berbentuk titik nikel pada ujung kontak.. Togel ini digunakan untuk mengontrol hampir semua komponen elektronik, sakelar togel telah diakui populer tetapi masih digunakan pada aplikasi asesoris otomotif.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.11 Sakelar DIP

Sakelar DIP adalah sebuah susunan sakelar kecil, sakelar yang terpisah, didisain untuk dipasang langsung pada papan rangkaian tercetak. Sakelar DIP mempunyai dua baris pin dengan jarak antar pin adalah 0.1 mil, danjarak baris pin adalah 0.3 mil yang tepat pada soket DIP standar.

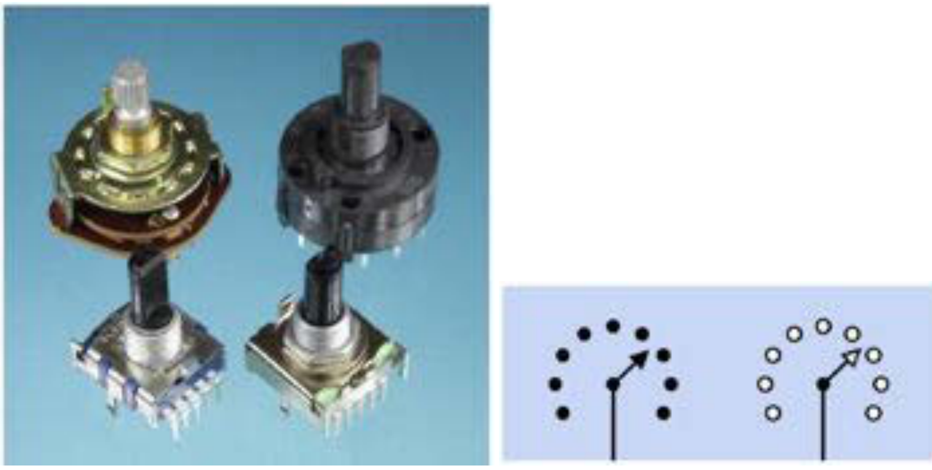


Gambar 2.13 Rangkaian Model Transfer Panas



2.6.12 Sakelar Rotary

Sebuah sakelar rotary membuat sebuah hubungan kelistrikan antara sebuah rotor, dipasang pada poros yang adalah ditekan oleh knop. Sebuah sakelar yang mempunyai banyak kutup, setiap hubungan dengan rotornya masing-masing. Rotor adalah seperti menjadi piringan terpisah dari sakelar, tetapi menunjuk pada arah yang berbeda, yang boleh dikombinasikan pada sebuah piringan tunggal, jika sakelar hanya mempunyai jumlah posisi sedikit.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.13 Sakelar Rotary DIP

Sebuah sakelar konvensional DIP adalah susunan array dari sakelar SPST miniatur yang didisain untuk tepat pada layout lubang DIP satndar. Sebuah sakelar rotary tidak cocok untuk layout lubang DIP. Dan sering dipakai pada panel secara terpisah.

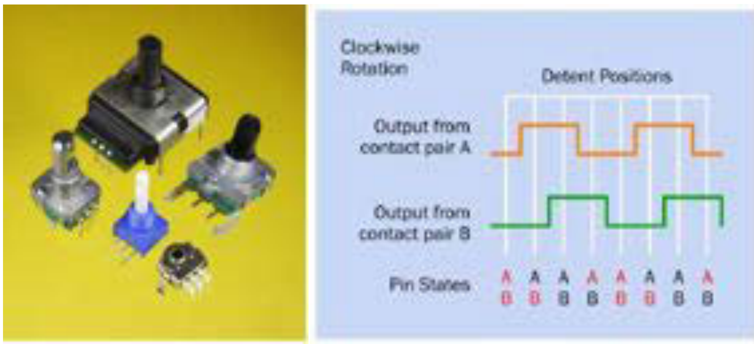


Gambar 2.13 Rangkaian Model Transfer Panas



2.6.14 Rotary Encoder

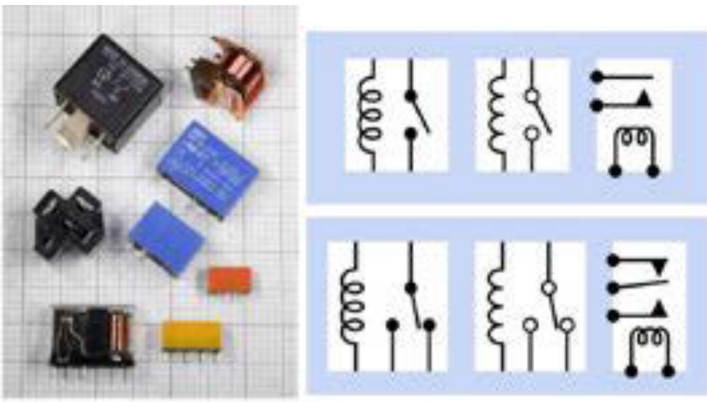
Sebuah rotary encoder yang mempunyai sebuah knop yang user dapat memutar untuk menampilkan secara seri pada prompt pada layar LCD, atau untuk mengatur input atau output seperti receiver stereo. Sebuah encoder mengandung dua pasang kontak, yang mana menghubung atau memutus keluaran phasa bila poros diputar. Dalam arah jarum jam, pasangan kontak A diaktivasi sebelum pasangan kontak B. Dalam arah berlawanan arah jarum jam , pasangan kontak B diaktivasi sebelum pasangan A.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.15 Relai

Sebuah relai terdiri sebuah kumparan, sebuah armatue dan sedikitnya satu pasang kontak. Arus listrik mengalir melalui kumparan, yang mana berfungsi sebagai elektromagnetik dan membangkitkan medan magnetik. Medan ini akan menarik armatur yang sering berbentuk sebagai bagian untuk menghubungkan atau memutuskan kontak.



Gambar 2.13 Rangkaian Model Transfer Panas



2.6.16 Potensiometer

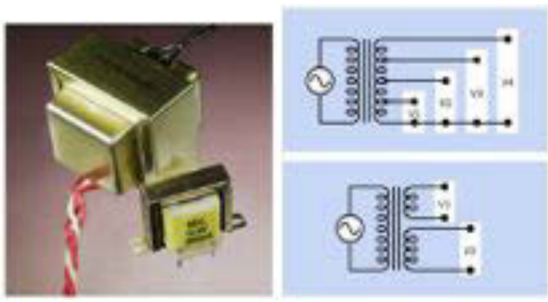
Sebuah potensiometer mempunyai tiga terminal. Kedua terminal terluar terhubung dengan ujung-ujung berlawanan dari sebuah elemen resistansi internal, seperti sebuah pelastik lempengan penghantar. Terminal pusat ketiga terhubung secara internal dengan sebuah kontak yang dikenal sebagi penyapu, dimana menyentuh strip dan dapat bergerak dari satu ke ujung yang lain melalui pemutaran poros atau penggeser.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.17 Transformator AC-AC

Arus listrik AC yang mengalir pada kumparan primer sebuah transformator mengiduksikan fluks magnetik dalam sebuah lapisan inti yang dibuat dari lempengan-lempengan plat besi.



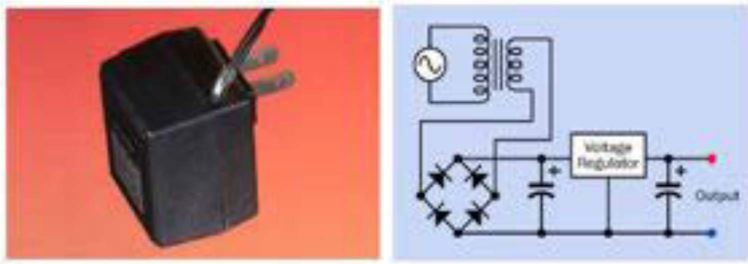
Gambar 2.13 Rangkaian Model Transfer Panas

Perubahan fluks menginduksi arus listrikpada kumparan sekunder, yang mana menyediakan keluaran arus AC pada output transformator. Besar tegangan yang ditransformasikan dari primer ke sekunder tergantung pada perbandingan jumlaj lilitan pada kumparan primer dan sekunder.



2.6.18 Power Supply AC-DC

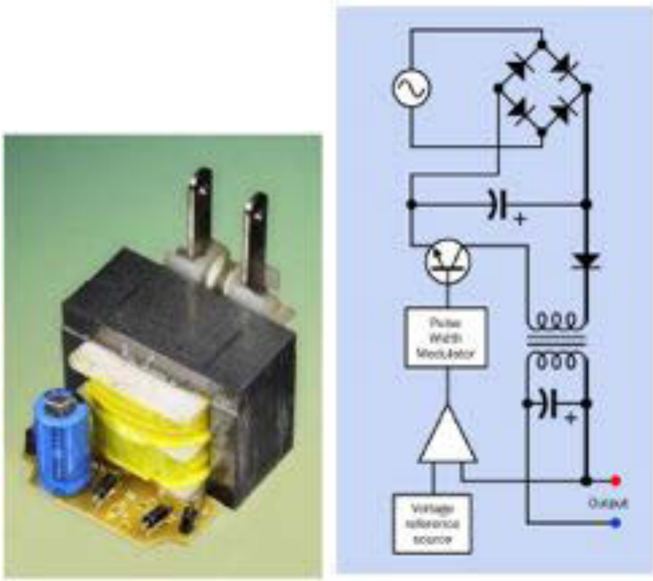
Jenis power supply dapat dijelaskan sebagai berbasis transformator, jadi langkah awalnya memiliki sebuah transformator yang dapat menurunkan tegangan AC pada sekundernya sebelum disearahkan. Karena penyearah dalam power suply secara umum melewatkan gelombang tegangan AC melalui sepasang diode silicon, akan terdapat drop tegangan 1.2V pada kedua diode tersebut. Kapasitor penghalus akan menyita tegangan sekitar 3V sebagai aksi untuk menghilangkan ripple, jadi output transfomator seharusnya sedikitnya 8VAC lebih tinggi dari tegangan yang diinginkan. Ini juga akibat daya akan hilangnya menjadi panas.



Gambar 2.13 Rangkaian Model Transfer Panas

2.6.19 Power Supply Switching AC

Jenis power supply dapat dijelaskan tanpa berbasis transformator.



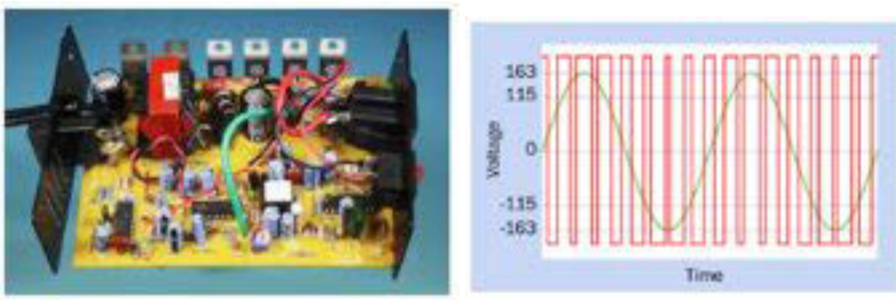
Gambar 2.13 Rangkaian Model Transfer Panas



Sebuah penyearah merubah input tegangan AC menjadi tegangan DC halus tanpa transformator. Sebuah konverter DC-DCsakelar tegangan DC on dan OFF pada sebuah frekuensi sangat tinggi menggunakan pulse width modulator (PWM) untuk mengurangi rata-rata tegangan efektif.

2.6.20 Power Supply Switching AC

Langkah pertama sebuah inverter menaikan teganagn 12VDC menjadi tegangan DC yang tinggi melalui sebuah internal converter DC-DC, kemudian menggunakan sebuah rangkaian pensakelaran untuk membuat sebuah pendekatan profil sinusoidal tegangan AC. Pensakelar digital menjaga untuk menghasilkan gelombang kotak, yang mana menggunakan pulse width modulation (PWM) untuk memberikan THD kurang dari 1%. Itu akan menghasilkan sebuah pulsa kotak yang frekuensinya lebih besar dari frekuensi output tegangan AC dan perubahan-perubahan lebar pulsanya akan membentuk gelombang sinusoidalmendekatitegangan efektif AC yang diinginkan.

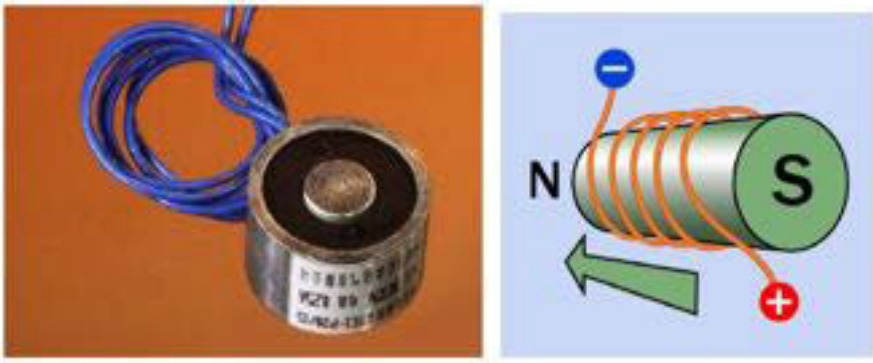


Gambar 2.13 Rangkaian Model Transfer Panas

2.6.21 Eletromagnet

Sebuah elektromagnet terdiri dari sebuahkumparan yang menghasilkan sebuah medan magnet dalam respon rangkaian listrik. Medan akan disalurkan dan diperkuat oleh inti bahan maknetik. Arus listrik mengalir melalui sebuah lingkaran kawat yang akan menginduksi sebuah medan maknet melalui pusat lingkaran kumparan. Jika sebuah potongan material ferromagnetic diletakkan pada pusat lingkaran kumparan, itu akan menghasilkan gaya magnetik. Kombinasi dari kumparan dan sebuah inti adalah sebuah elektromagnetik.





Gambar 2.13 Rangkaian Model Transfer Panas

2.6.22 Selenoid

Arus listrik mengalir melalui kumparan menghasilkan sebuah gaya magnetik. Jika sebuah batang yang dibuat dari besi lunak, diletakkan pada pusat lingkaran kumparan, kumparan akan menginduksikan sebuah polaritas magnetik yang sama dan berlawanan pada besi lunak tersebut. Akibatnya batang besi lunak akan tetarik ke posisi dalam kumparan. Apabila batang besi lunak semakin masuk berada pada pusat lingkaran, maka gaya elektromagnetiknya akan bertambah, sehingga daya tarik magnetik semakin kuat..

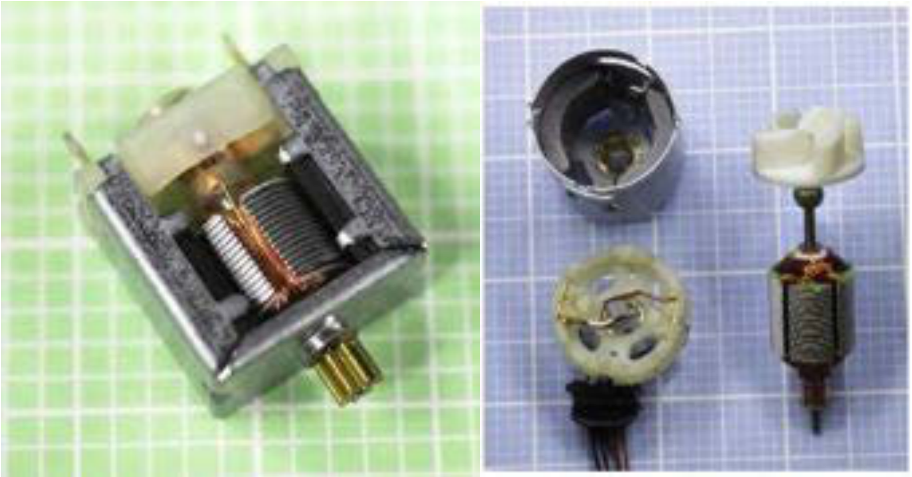


Gambar 2.13 Rangkaian Model Transfer Panas

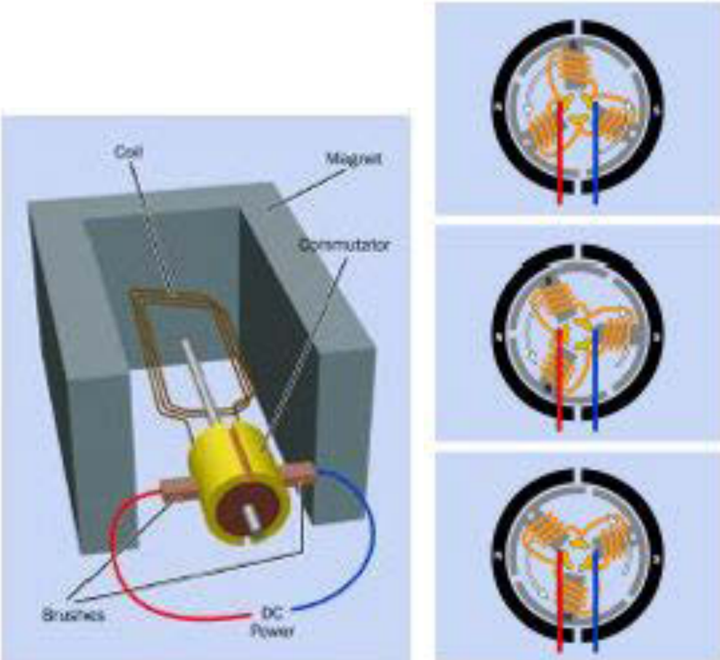


2.6.23 Motor DC

Arus listrik mengalir melalui dua atau lebih kumparan yang terpasang pada poros motor dan memutar itu, ini disebut rotor.



Gambar 2.13 Rangkaian Model Transfer Panas



Gambar 2.13 Rangkaian Model Transfer Panas

Gaya magnetik dihasilkan oleh arus listrik yang dikonsentrasikan melalui inti atau kutup dari besi lunak dan berinteraksi dengan medan magnetik yang dihasilkan oleh magnet permanen yang dipasang sekeliling rotor, ini disebut stator. Daya pada kumparan dikirim melalui sepasang sikat, sering dibuat dari kompon graphit. Pegas menekan sikat pada sebuah pembagi yang berputar



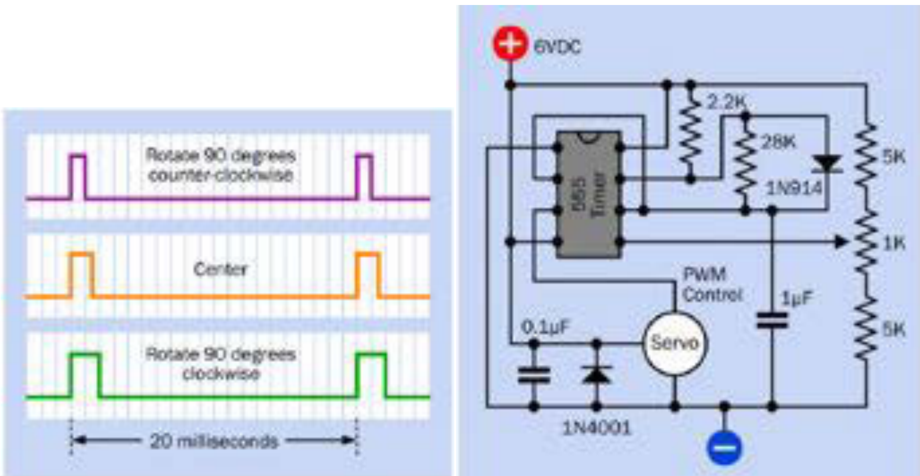
mengikuti poros dan terbagi menjadi beberapa bagian dan terhubung dengan kumparan, ini disebut komutator. Saat komutator berputar, bagian-bagiannya memberikan tegangan melalui sikat ke kumparan stator, dalam sebuah pensakelaran mekanis.

2.6.24 Motor Servo

Sebuah motor servo adalah sebuah kombinasi dari sebuah motor, gir pereduksi putaran dan kontrol elektronik miniatur, biasanya dikemas bersama-sama didalam kemasan plastik yang kompak. Motror itu sendiri bisa motor DC atau AC. Motor servo biasanya dikontrol melalui pulse width modulation (PWM). Skema encoder dari sinyal kontrol pada sebuah tugas berat servo, didisain untuk berjalan pada tegangan tertentu.



Gambar 2.13 Rangkaian Model Transfer Panas

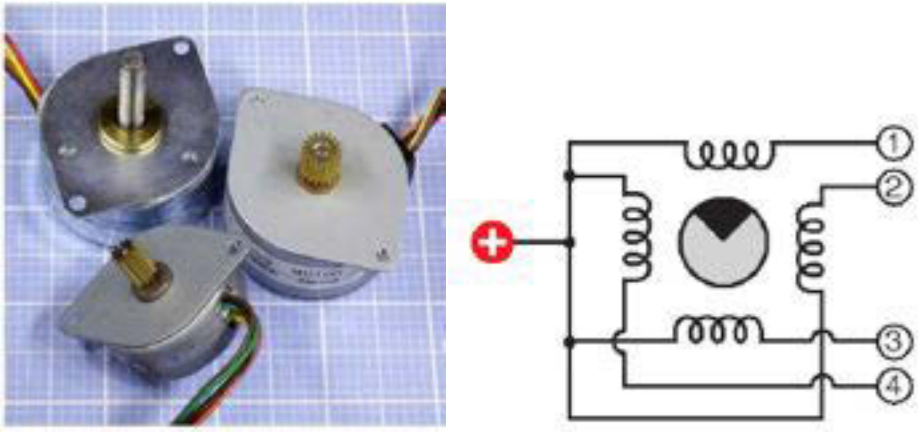


Gambar 2.13 Rangkaian Model Transfer Panas

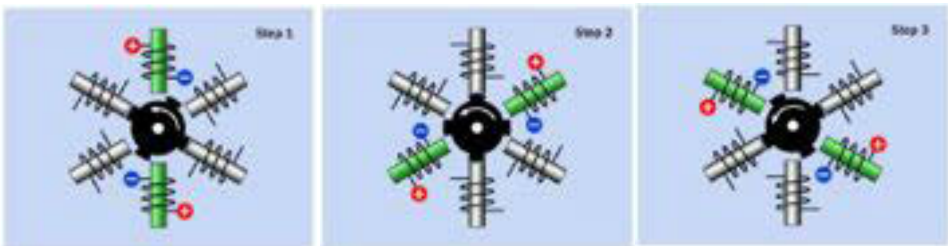


2.6.25 Motor Stepper

Stator mempunyai multi kutup dari besi lunak. Setiap kutup apakah dienegikan oleh kumparan sendiri atau bersama-sama. Pada semua jenis motor stepper, kutup-kutup staor dimaknetisasi secara sekuensial untuk memutar rotor dan dapat dimagnetisasi ulang dalam satu konfigurasi untuk menjaga satsionari putaran rotor. Rotor terdiri sebuah atau banyak magnet permanent, dimana berinteraksi dengan medan magnet yang dihasilkan dalam stator.



Gambar 2.13 Rangkaian Model Transfer Panas

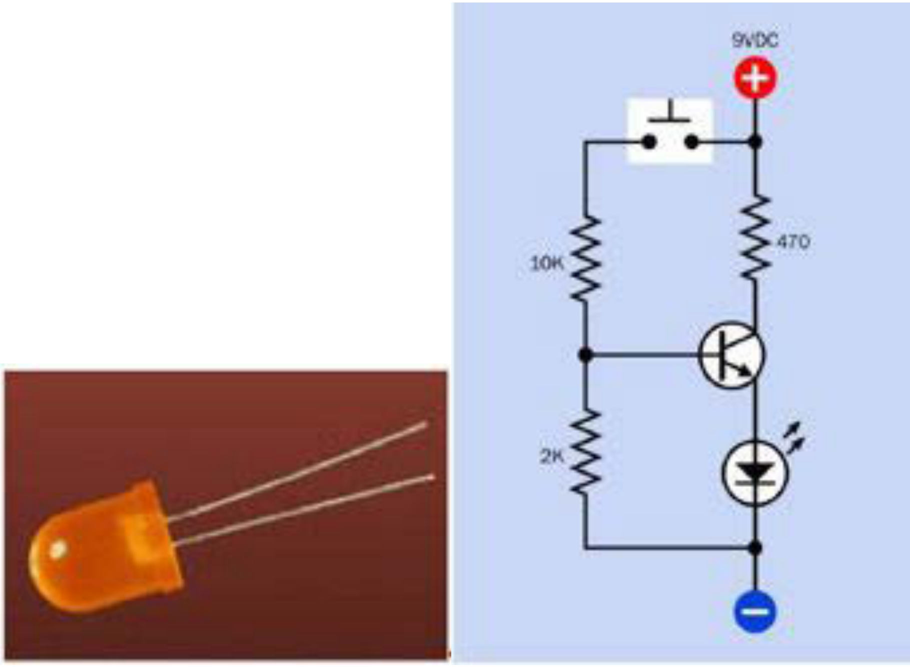


Gambar 2.13 Rangkaian Model Transfer Panas



2.6.26 LED

Operasi photodioda dapat juga dibalik dengan bias maju dioda dan meyebabkan sebuah level rekombinasi yang signifikan untuk mengambil tempat dalam region depleksi. Beberapa energi akan delepaskan dan dikonversi menjadi energi cahaya oleh emisi photon. Jadi dioda yang bekerja pada mode ini akan menghasilkan cahaya bila dibias maju. Photodioda yang digunakan dalam cara ini disebut Light Emitting Diode (LED).



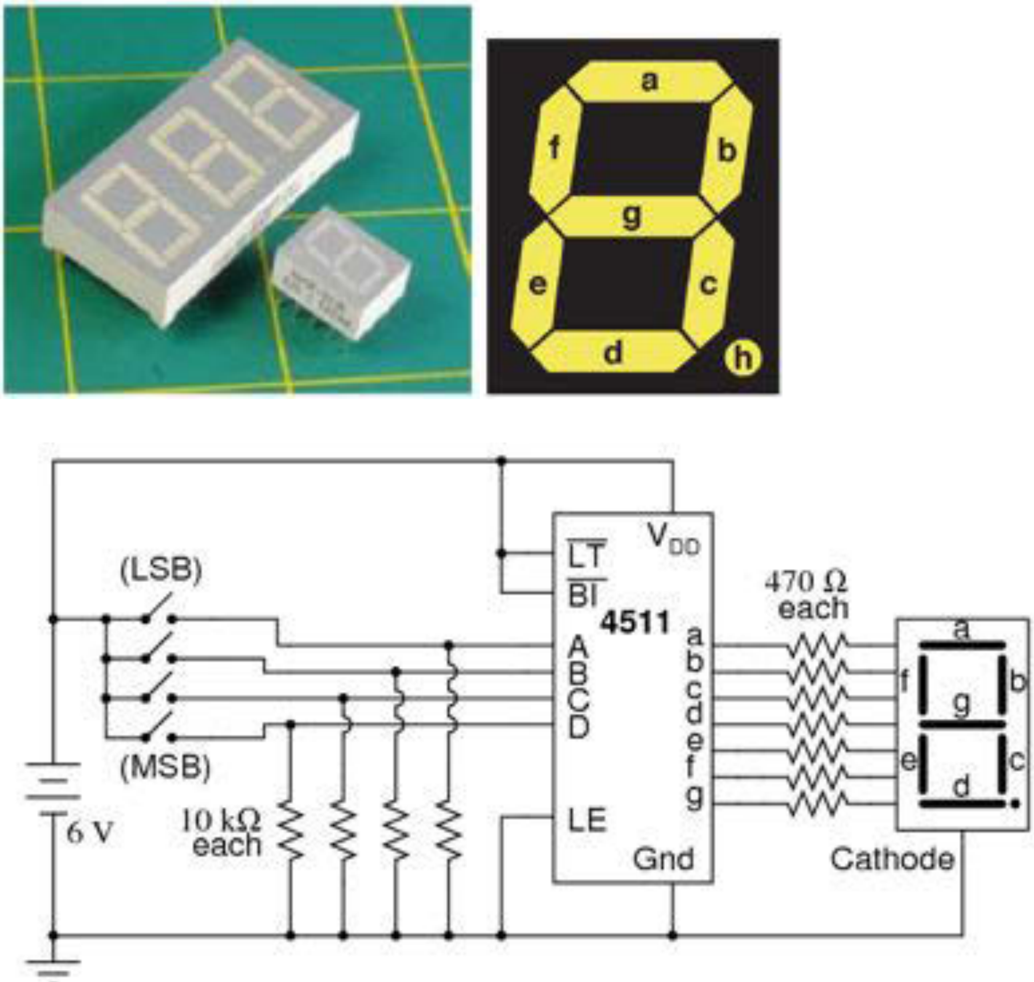
Gambar 2.13 Rangkaian Model Transfer Panas

2.6.27 Seven Segment

Sebuah penampil seven segment adalah pengelompokan light emitting diodes (LED) yang tersusun dalan pola tertentu. Delapan LED disusun dalam pola angka delapan, ini disebut seven segment. Semua segmen dapat membentuk bilangan dari o sampai 9. Penampil seven segment adalah komponen yang sering dipakai pada rangkaian digital.. sangat baik untuk memahami rangkaian drivernya dan IC 4511 adalah jenis driver yang baik sebagai drivernya.



Prinsip operasinya adalah memberikan input sebuah Bila Its operating principle is to input a four-bit BCD (Binary-Coded Decimal) value, and energize the proper outinput BCD 4 bit untuk membentuk digit desimal pada seven segmen yang sesuai imput BCD. Input-input BCD didisain A, B, C, dan D dengan urutan dari least-significant byte (LSB) ke most-significant byte (MSB). Outputnya adalah dilabel a, b, c, d, e, f, dan g, setiap hurup berhubungan pada segmen standar seven segment. Jadi setiap segmen LED diperlukan pemasangan seri resistor padanya, kita gunakan resistor 470 Ω antara output 4511 dan terminal-terminal yang sesuai dengan LED seven segmen.



Gambar 2.13 Rangkaian Model Transfer Panas



2.6.28 Buzzer

Buzzers adalah sebuah komponen elektronik yang biasa digunakan untuk menghasilkan suara. Ringan, sederhana konstruksinya dan harganya murah membuat itu sangat berguna bagi berbagai aplikasielektronik seperti indicator mobil saat mundur, computer dan alain-lain. Piezo buzzer bekerja berdasarkan pada kebalikan dari prinsip piezo elektrik, yaitu fenomena membangkitan listrik bila terdapat tekanan mekanik pada logam tertentu dan Material seperti itu disebut material piezo elektrik. Piezo keramikadalah material, yang membangkitkan efek piezo electric dan secara luas digunakan buzzer. Bila diarahkan ke sebuah medan magnet listrik bolak balik, mereka akan tertarik dan tertekan sesuai frekuensi sinyal sehingga menghasilkan suara.



Gambar 2.13 Rangkaian Model Transfer Panas



Rangkuman 1

Latihan 1

Tugas 1

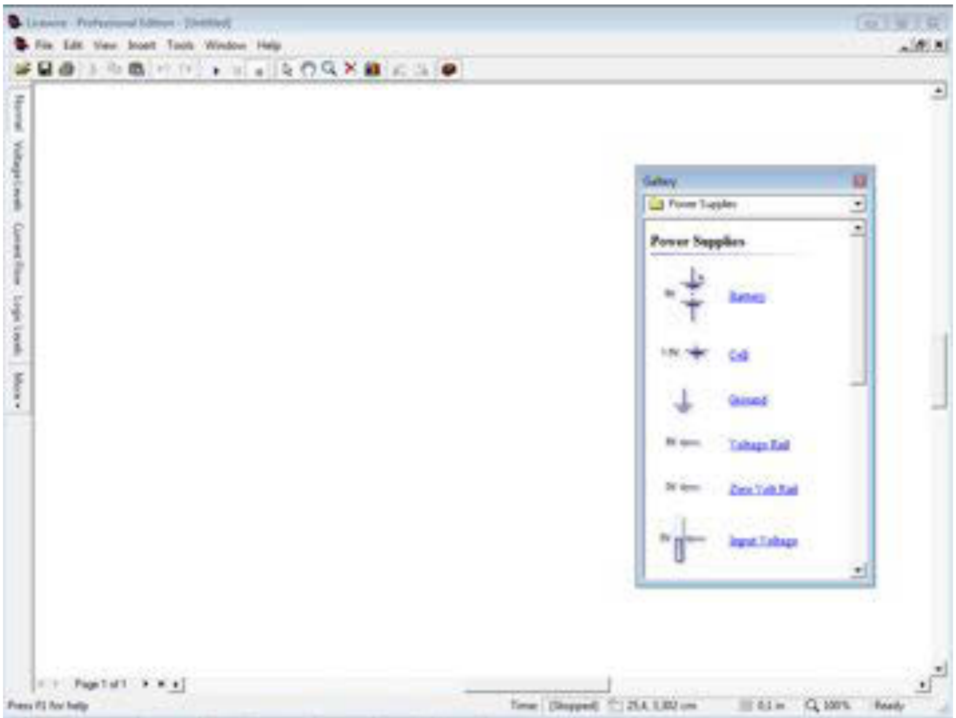
Kunci Jawaban 1



KEGIATAN BELAJAR 3

3.1 Mengenal Livewire

Livewire adalah suatu program yang merupakan suatu simulasi elektronika yang digunakan untuk merancang dan untukanalisa, ditampilkan dalam bentuk animasi untuk mempertunjukan fungsi atau prinsip dasar dari rangkaian elektronika. Program *Livewire* termasuk perangkat lunak aplikasi yang merupakan suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Jenis perangkat lunak yang digunakan adalah program *Livewire-Professional Edition* versi 1.11, dapat dilihat di http://www.newwave-concepts.com merupakan program yang berlisensi (*License-Ware*), perangkat lunak yang dilindungi oleh hukum hak cipta.



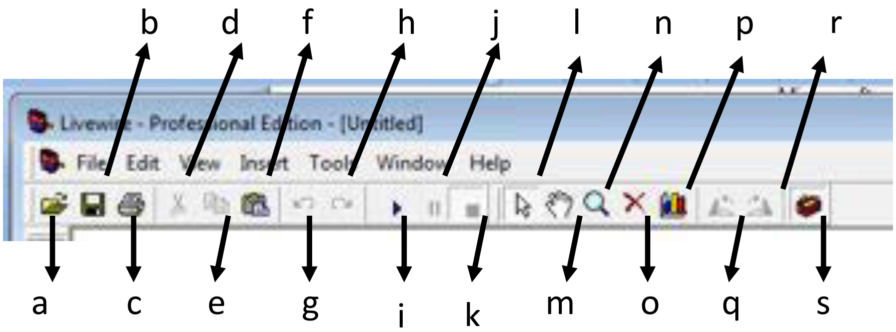
Gambar 3.1. Tampilan Livewire saat awal dibuka



Jalankan program *Livewire-Professional Edition* versi 1.11, untuk membuat skematik rangkaian elektronika. Setelah dijalankan, akan muncul window Livewire – Professional Edition.

Pada Program Livewire terdapat beberapa menu pilihan mulai dari menu file,edit,view, insert, tools, window maupun help. Menu-menu ini memiliki kemiripan dengan aplikasi lain yang dibuat seperti Microsoft yang sudah dahulu dipahami, dengan beberapa diantaranya merupakan menu khusus yang didesain untuk Livewire itu sendiri. Menu file hamper sama dengan format menu pada program-program rancangan Microsoft atau rancangan program lain, yang terdiridari sub menu New, Open, Close, Save, Save as, Protect Document, Preview in Browser, Page Setup, Print, Sent..., Properties dan Exit.

3.2 Fungsi Toolbar



Gambar 3.2 Fungsi toobar

a) Open – perintah ini digunakan untuk membuka dokumen atau data dengan ekstensi *.lvw atau Livewire Document.

b) Save – perintah ini digunakan untuk menyimpan dokumen yang berada di lembar kerja. Dalam penyimpanan capture akan menciptakan backup dengan ekstensi *.lvw atau Livewire Document.

c) Print – perintah ini digunakan untuk mulai mencetak halaman skematik yang aktif atau Sitem- Sitem yang telah dipilih dalam project manager.

d) Cut – perintah ini digunakan untuk menghapus objek yang dipilh dari window dan menempatkannya di Clipboard. Perintah ini hanya aktif bila ada objek yang dipilih.



e) Copy – perintah ini digunakan untuk menduplikasikan objek yang dipilih dan mengirimnya ke Clipboard. Hanya aktif bila ada objek yang dipilih.

f) Paste – perintah ini digunakan untuk memindahkan objek yang disimpan di Clipboard ke window yang aktif. Perintah ini tidak berfungsi bila Clipboard dalam keadaan kosong.

g) Undo – perintah ini digunakan untuk membatalkan efek perintah sebelumnya.

h) Redo – perintah ini digunakan untuk membatalkan efek perintah Undo.

i) Run – perintah ini digunakan untuk menjalankan simulasi rangkaian yang telah jadi pada Clipboard.

j)Pause – perintah ini digunakan untuk menghentikan atau menunda sementara pada proses rangkaian yang sudah dijalankan. Perintah ini masih menunjukan hasil analisis pada rangkaian saat terakhir perintah ini dijalankan.

k) Stop – perintah ini digunakan untuk menghentikan secara permanen pada proses rangkaian.

l)Select Tool – perintah ini digunakan untuk memilih objek.

m) Pan Tool – perintah ini digunakan untuk menggeser Clipboard.

n) Zoom Tool – perintah ini digunakan untuk mengubah tampilan menjadi besar dan kecil pada objek.

o)Delete Tool – perintah ini digunakan untuk menghapus objek yang dipilih.

p) Graph – perintah ini digunakan untuk menampilkan grafik.

q) Rotate Left – perintah ini digunakan untuk memutar ke kiri objek yang dipilih dengan kenaikan 90 derajat.

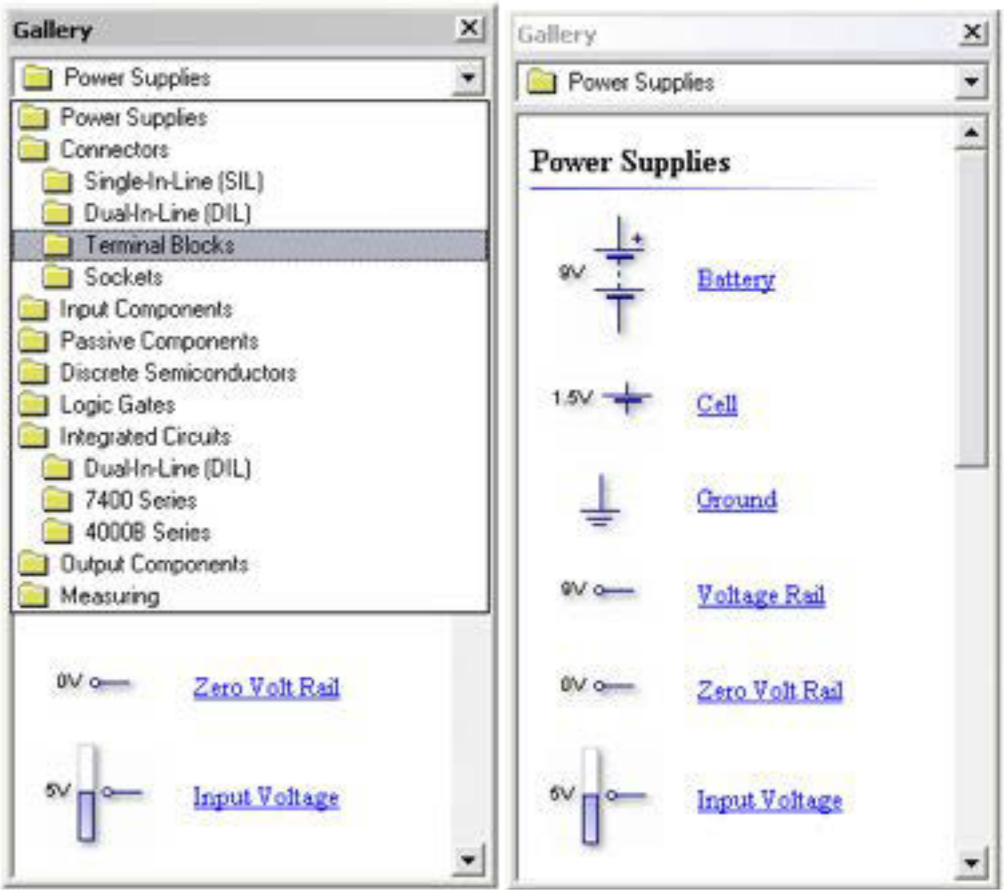
r) Rotate Right - perintah ini digunakan untuk memutar ke kanan objek yang dipilih dengan kenaikan 90 derajat.

s) Gallery – perintah ini digunakan untuk memunculkan jendela Gallery dengan berbagai macam komponen yang tersedia.



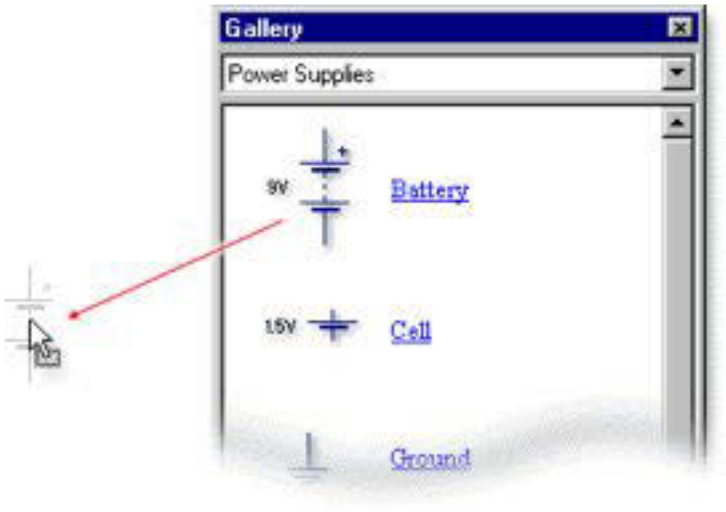
3.3 Langkah Kerja

Dari menu Gallery, dapat memilih jenis komponen yang akan ditempatkan pada lembar kerja *Livewire* atau *stage*. Klik **Gallery** (CTRL+F2) dan pilih jenis komponen yang diinginkan dan klik sambil tahan shortcut atau simbol dari komponen tarik ke *stage*



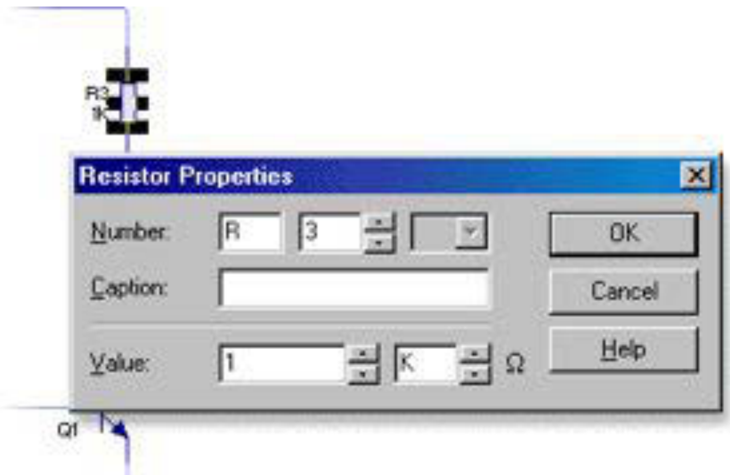
Gambar 3.3 Tampilan pemilihan komponen

Dari simbol komponen dapat dirubah nilai, nama, no tergantung dari jenis komponen. Klik 2 kali pada komponen dan muncul jendela "**Cell Properties**" tergantung dari jenis komponen yang dipilih



Gambar 3.4 Gallery

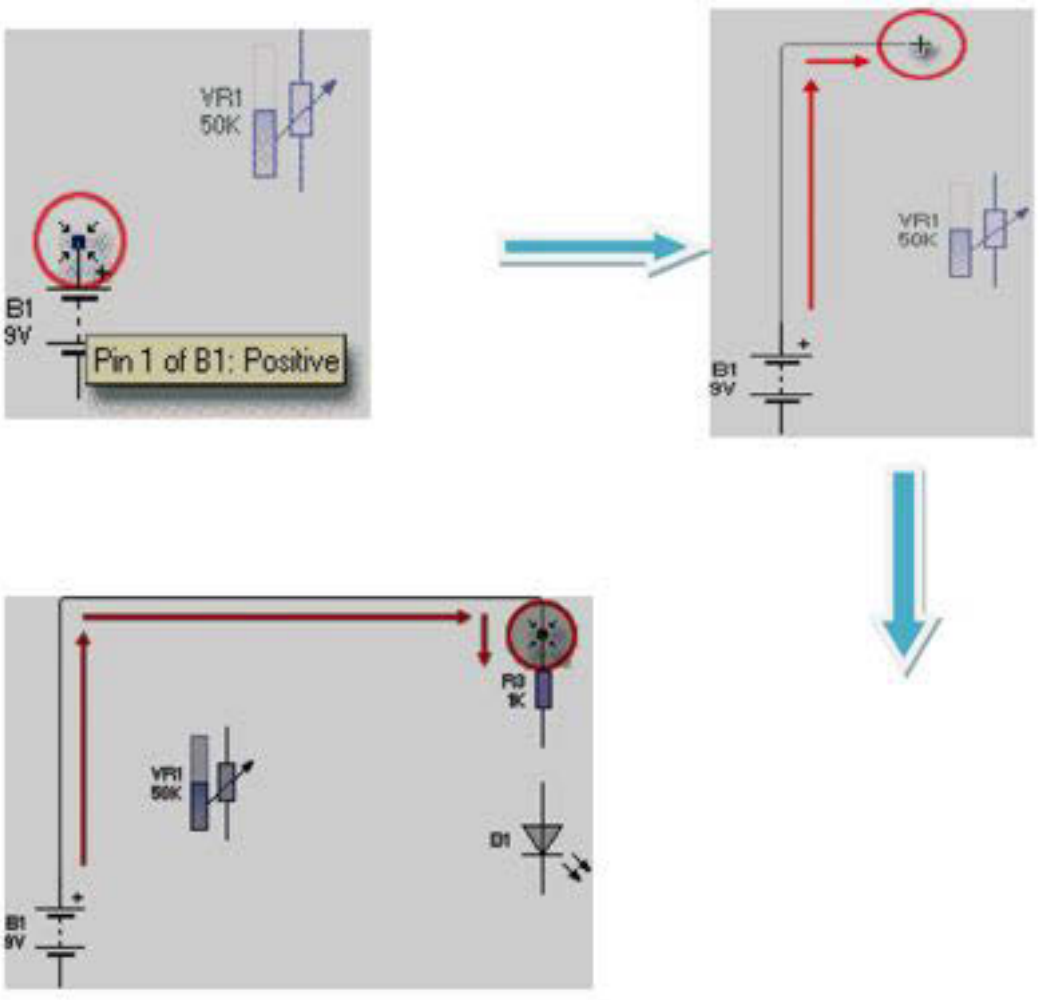
Pemilihan kompenen dari gelery dengan cara double klik komponen yang akan dipilh dan kompenen tersebut akan muncul di area kerja dan kita harus meposisikan kompenen yang dipilih kearea kerja sesuai dengan rangkain yang kita inginkan.



Gambar 3.5 Cell properties

Dari simbol komponen dapat dirubah nilai, nama, no tergantung dari jenis komponen. Klik 2 kali pada komponen dan muncul jendela ”**Cell Properties**”tergantung dari jenis komponen yang dipilih. Kemudian untuk

memulai suatu suatu rangakaian maka terlebih dahulu dipilih komponen komponen rangkaian dari **menu Gallery**. Dan double klik kaki komponen untuk mengubungkan masing- masing rangkaian dengan wire, tahapanya akan dijelasakan pada gambar berikut:



Gambar 3.6 Tahap merangkai dengan livewire

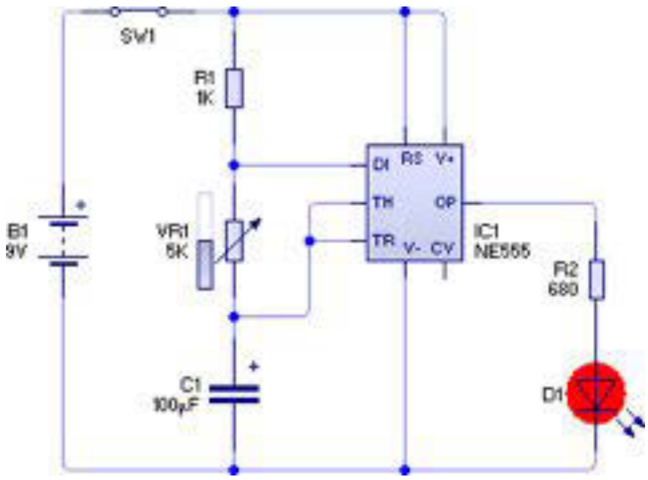
Komponen-komponen yang disusun dapat saling dihubungkan menggunakan Wire. Klik di ujung simbol komponen yang diinginkan untuk memulai pengawatan (berubahnya cursor menjadi tempat yang menunuk ke Pin) tarik mouse hingga ke kaki komponen yang diinginkan. Untuk rangkaian yang sudah tersusun di area kerja dapat dijalankan dengan memilih toolbar menu kemudian mengklik menu “**Run”** atau dengan menakan tombol **CTRL + F9** untuk **STOP** dan kemudian **F9** untuk **Run** lagi seperti yang dijelaskan pada fungsi menu toolbar sebelumnya



3.4 Menggambar dan menganalisa IC Timer 555

Pada contoh pembelajaran ini, siswa diharapkan mampu untuk menggambar dan mensimulasikan rangkaian elektronik dengan menggunakan LiveWire. Untuk itu ikutilah petunjuk tutorial keterampilan dasar untuk menggambar rangkaian elektronik dengan menggunakan Livewire secara efektif.

Rangkaian leketronik yang akan dipelajari adalah rangkaian elektronik IC Timer 555 yang di jalankan dalam mode *astable*. Rangkaian tersebut akan membuat lampu LED yang terpasang berkedip "ON" dan "OFF" seperti tampak pada gambar di bawah ini.



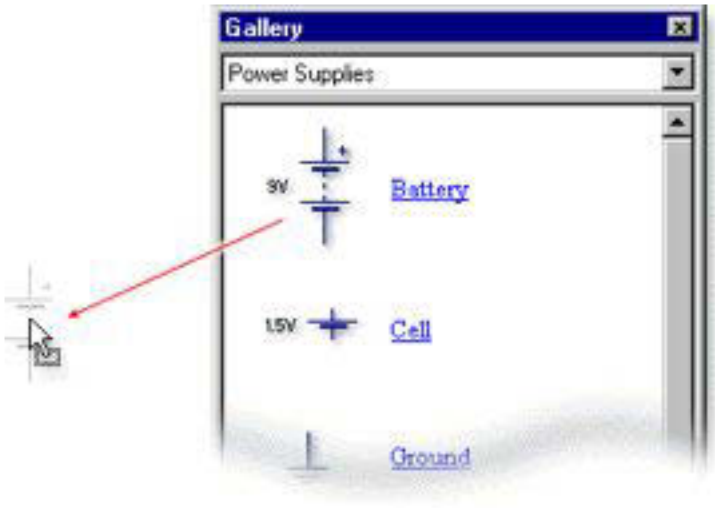
Gambar 3.7Rangkaian *astable* IC Timer 555

Untuk menggambar rangkaian di atas, dimulai dengan membuka aplikasi Livewire hingga muncul lembar halaman kerja baru (*empty document*). Selanjutnya tambahkan komponen dengan menggunakan menu "*Gallery*". Jika menu "Gallery" tidak muncul, klik-lah tombol "Gallery" yang terdapat pada toolbar paling atas  .

Setelah gallery komponen terbuka, maka akan terlihat kelompok komponen yang akan diperlukan untuk menggambar rangkaian di atas. Pengelompokan komponen berdasarkan fungsi komponen tersebut.



Pada kelompok ***Power Supplies***, tambahkan sebuah Battery ke dalam halaman kerja dengan cara menggeret komponen (tekan dan tahan tombol mouse kiri) dari gallery kehalaman kerja seperti tampak pada gambar berikut.



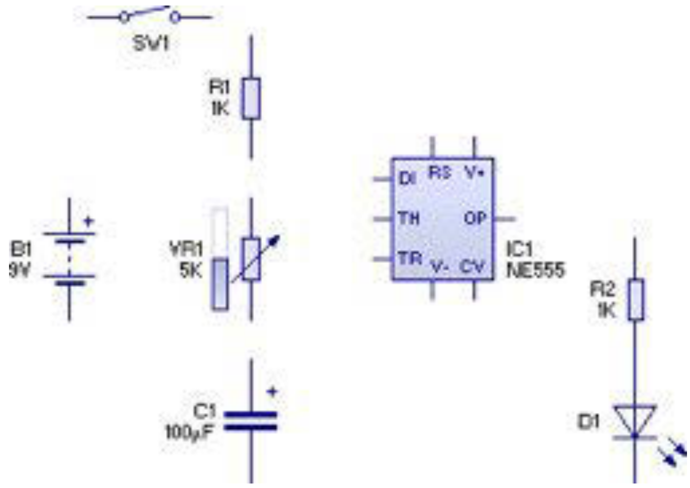
Untuk membuat rangkaian IC Timer 555, di butuhkan beberapa komponen lain. Tambahkan komponen berikut kedalam halaman kerja :

- SPST Switch (1 buah)pada pada kelompok input komponen ;
- Variable Resistor (1 buah) pada pada kelompok input komponen ;
- Resistor (2 buah) (1 bauah) dari kelompok komponen pasif ;
- electrolytic Capacitor (1 bauah) dari kelompok komponen pasif ;
- Timer 555 (1 buah) dari kelompok Ics (Analogue/Mixed);
- LED (1 buah) dari kelompok komponen output;

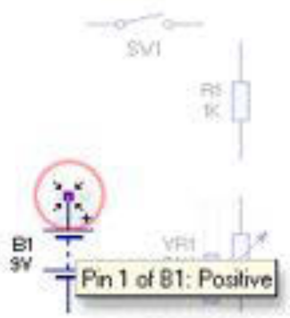
Komponen- komponen tersebut dapat di atur posisi peletakannya dengan memilih select button (tanda panah) pada toolbar , sehingga pointer pada layar akan tampak seperti standar pointer 

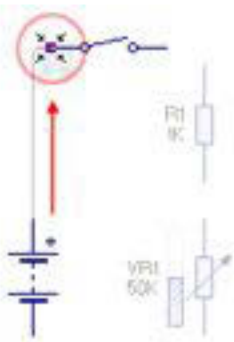


Untuk memudahkan merangkai, sebaiknya letakan komponen saling berdekatan sebelum menghubungkan tiap komponen dengan wire/kabel seperti tampak pada gambar berikut.



Langkah selanjutnya adalah menghubungkan tiap-tiap komponen dengan wire/kabel sehingga membentuk suatu rangkaian Timer 555 astable. Untuk melakukan hal tersebut, pertama-tama arahkan kursor pada ujung pin atas dari batterai, sehingga muncul keterangan kecil yang mendiskripsikan komponen batterai tersebut (gambar a). Kemudian tekan dan tahan tombol kiri pada mouse, geretlah mouse hingga muncul garis wire/kabel dan hubungkan ke komponen switch SW1 kemudian lepaskan tombol kiri mouse sehingga tampak seperti pada gambar b.

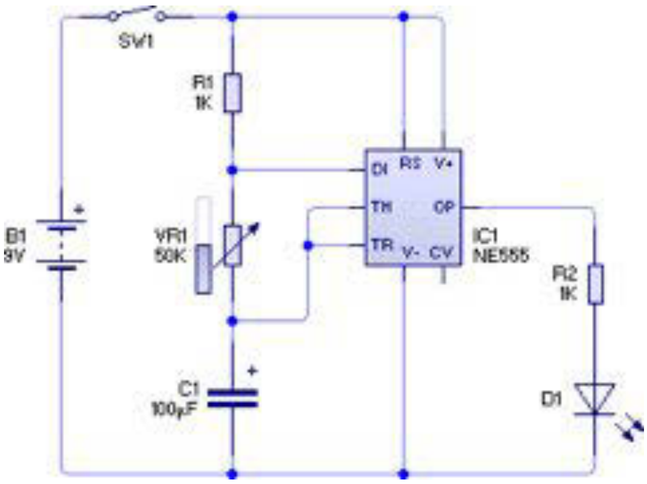




Gambar a   Gambar b



Selanjutnya hubungkan tiap komponen secara keseluruhan seperti pada gambar dibawah ini sebgai acuan rangkaian. Jika anda menemukan kesulitan, maka anda dapat melakukan fungsi undo untuk melakukan koreksi kesalahan.



Setelah rangkaian terhubung seperti di atas, langkah selanjutnya adalah merubah nilai spesifikasi tiap komponen. Pengaturan waktu rangkaian timer 555 astable ditentukan oleh dua resistor dan sebuah kapasitor. Pada rangkaian ini diwakili oleh **R1, VR1** dan **C1.**

Pengaturan lama waktu lampu LED akan aktif "ON" ditentukan persamaan berikut :

$$f = \frac{1.44}{(R1 + 2VR1) \times C1} \qquad \text{...(1)}$$

Dimana :

R1 = 1K Ω ;

VR1 = 50K Ω(50% dari 100K ketika slider di posisi tengah) ;

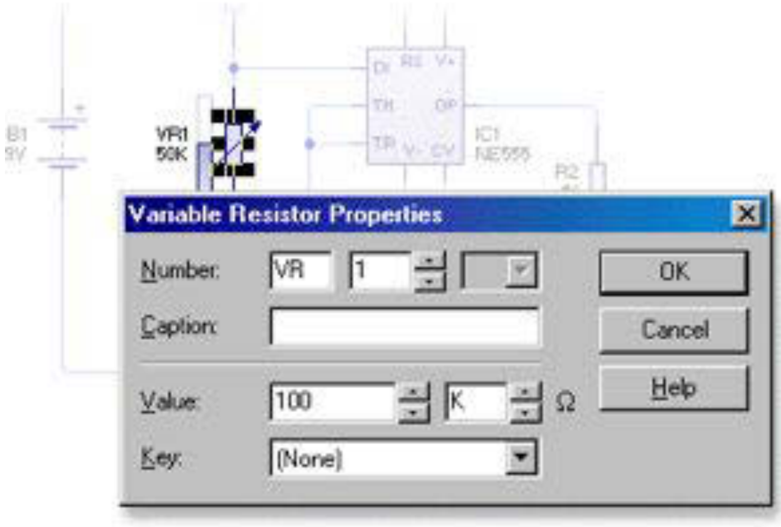
C1 = 100µF ;

Dari perhitungan frekuensi (f) didapatkan 0.14 Hz (Hertz) sehingga LED akan menyala berkedip setiap 7 detik (periode = 1/frekuensi).

Untuk menyalakan lampu LED berkedip lebih cepat, maka variabel resistor 100K dapat diganti 10K. Gunakan rumus perhitungan di atas untuk mengkalkulasi nilai periode LED untuk berkedip.



Untuk mengganti nilai properti dari variabel resistor VR1, maka double-klik pada komponen VR1 sehingga muncul jendela properties variabel resistor properties seperti pada gambar berikut.



Nilai variabel resistor VR1 ditunjukkan pada kolom “Value” dan desebelah kanannya terdapat kolom multiplikasi. Nilai maksimal variabel resistor (dalam ohm) di hitung dan dikalikan dengan nilai multiplikasinya. Perhatikan contoh berikut :



Pada rangkaian timer 555 astabel diatas, kita hendak merubah nilai variabel resistor dari 100K Ω menjadi 10K Ω. Untuk itu pada kolom value masukan angka “10” dan pilih nilai “K” (kilo=1000) pada kolom multiplikasinya.

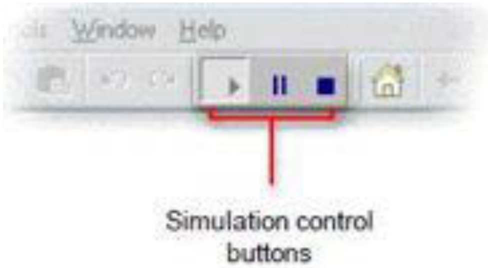
Langkah terakhir yang harus dilakukan adalah mengganti nilai dari resistor R2 menjadi 680 Ω. Double-klik komponen resistor R2dan ubahlah nilai value menjadi 680 dan ubah kolom multiplikasi “K” (x1000) menjadi kosong (x1). Resistor R2 berfungsi sebagai pembatas arus rangkaian yang melewati LED.



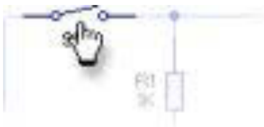
Jika sumber baterai yang digunakan 9Volt, dengan memasang nilai resistor 680 Ω, diharapkan arus yang melewati LED kurang lebih 10mA (mili amper).

3.5 Simulasi rangkaian

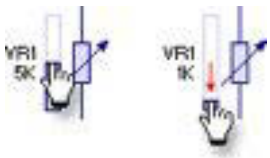
Setelah rangkaian timer 555-astable telah terangkai pada halaman kerja, maka selanjutnya kita dapat melakukan simulasi kinerja dari rangkaian tersebut. Pertama-tama klik tombol RUN yang terdapat pada toolbar



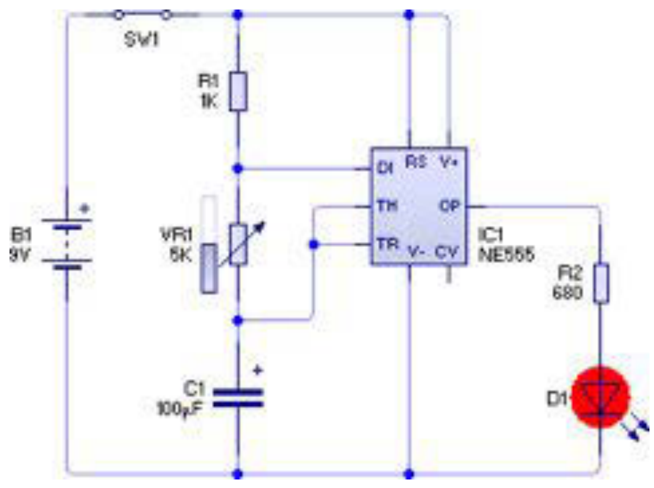
Ketika tombol RUN ditekan, terlihat seperti tidak terjadi perubahan pada rangkaian. Hal ini kare tombol ON/OFF yang mengaktifkan rangkaian belum bekerja. Selanjutnya klik kiri mouse pada komponen switch SW1 sehingga rangkaian bekerja.



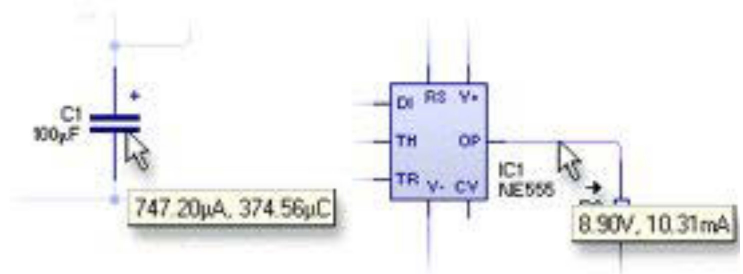
Setelah tombol SW1 diaktifkan maka rangkaian akan bekerja dan terlihat bahwa lampu LED akan menyala berkedip ON-OFF dengan perioda waktu berdasar pengaturan variabel resistor VR1. Jika kita ingin mengatur kecepatan kedip dari lampu LED, maka variabel resistor harus di atur dengan mengadjust slider. Untuk mengadjust slider maka letakan krusor pada komponen variabel resistor, klik tombol kiri mouse dan tahan kemudian geretlah posisi slider naik atau turun sehingga nilai dari variabel resistor akan ikut berubah seperti gambar berikut.







Kita dapat mengetahui nilai arus yang mengalir dan tegangan pada tiap titik yang hendak kita ukur dengan menggunakan fasilitas *pop-up hints*. Untuk menggunakan fasilitas pop-up hints arahkan kursor ke arah wire/kabel yang hendak kita selidiki nilai tegangan atau arusnya. Hasil pop-up hints akan tampil seperti gambar di bawah ini.

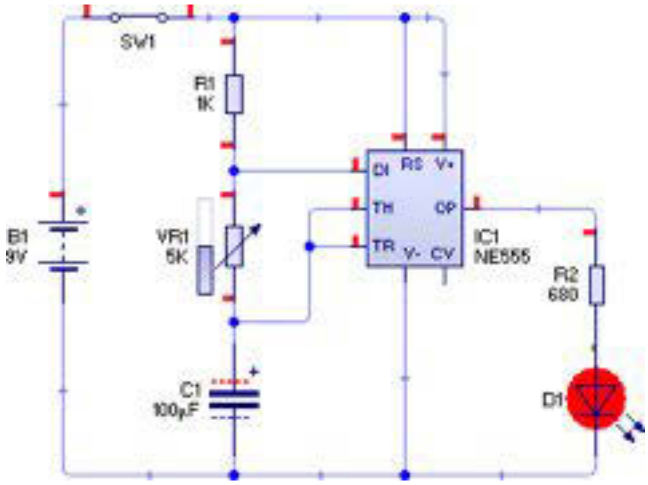


Fungsi lain simulasi dari Livewire adalah fungsi animasi yang merepresentasikan konsep elektronik yang terjadi dalam rangkaian tersebut seperti nilai tegangan, arus, pengisian, pengosongan, aliran arus, nilai digital dan lain sebagainya.

Fungsi simulasi animasi yang disediakan pada livewire terdapat empat mode operasi yaitu normal, Voltage level, Current Flow dan Logic Level. Keempat mode operasi tersebut terletak di panel jendela sebelah kiri.



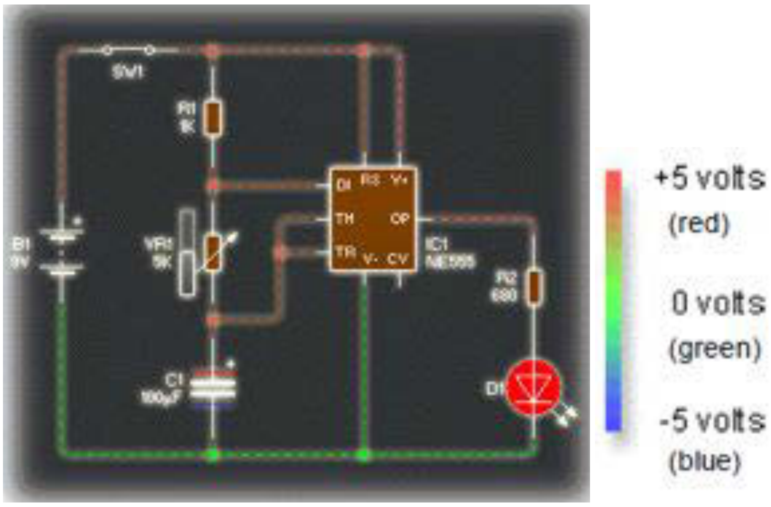
Ketika simulasi dijalankan pada mode Voltage Level, maka simulasi rangkaian akan tampak seperti gambar di bawah ini. Tampak pada gambar tersebut kotak indikator kecil berwarna merah dan biru pada setiap jalur komponen. Indikator tersebut mereprensetasikan perbedaan level tegangan dan tanda panah yang menunjukan arah dari aliran arus.



Pada simulasi animasi *Voltage Level* rangkaian elektronik IC Timer 555 terdapat indikator yang menunjukan nilai pengisian kapasitor. Setiap fungsi tanda "+" dan "-" menunjukkan tingkat pengisian dan pengosongan senilai 100µC (coloumb).

Selanjutnya pilih mode simulasi Current Flow yang terdapat pada jendela sebelah kiri dari aplikasi Livewire. Pada mode ini akan ditampilkan arah aliran arus yang dianimasikan dengan pergerakan garis putus-putus pada sepanjang wire/kabel.

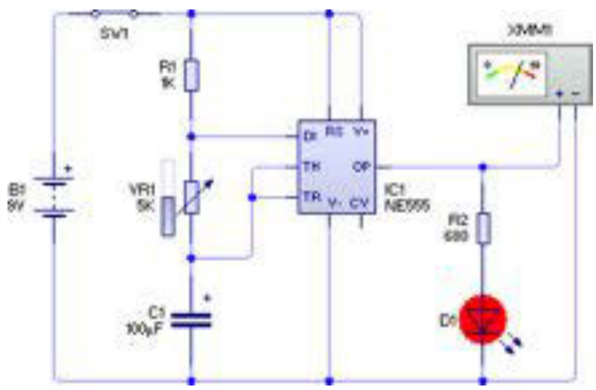
Disamping itu juga level tegangan ditampilkan berdasar gradasi warna. Warna merah untuk level tegangan 5 volt atau lebih, warna biru untuk -5 volt atau kurang dan warna hijau untuk merepresentasikan nilai 0 volt (ground).



3.6 Melakukan pengukuran pada rangkaian Livewire

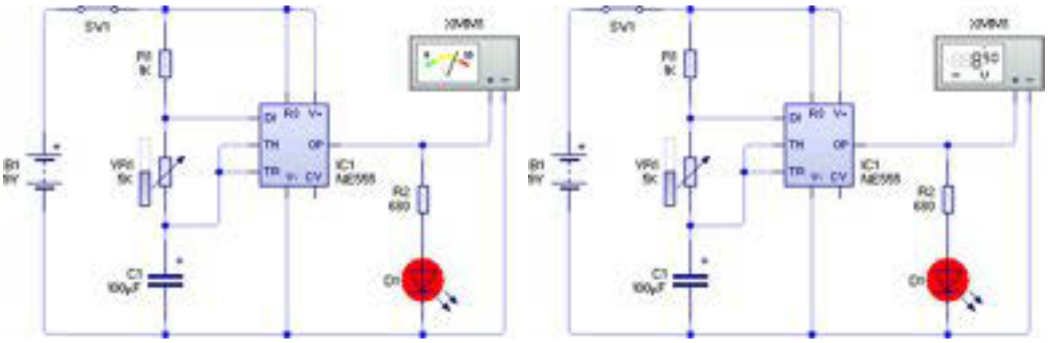
Pengukuran rangkaian pada livewire mengacu pada kaidah pengukuran alat-alat listrik seperti dunia nyata. Artinya dalam pengukuran harus di perhatikan dimana letak posisi alat ukur, kabel positif dannegatif dari alat ukur, posisi ukur pararel untuk mengukur tegangan, posisi seri untuk mengukur arus dan lain-lain.

Untuk menambahkan komponen alat ukur, maka ubah posisi running menjadi stop. Kemudian tambahkan komponen ***Analogue Multimeter*** dari kelompok ***Measuring***. Komponen ini memiliki dua buah pin, pin positif (+) dan pin negatif (-). Hubungkan pin positif (+) ke titik pengukuran resistor R2 680Ω dan hubungkan pin negatif (-) ke ujung bawah lampu LED (titik ground).



Tekan tombol RUN untuk menjalankan kembali simulasi Livewire, perhatikan jarum pada alat ukur meter akan berubah menjadi sinyal output perubahan timer (LED). Seperti pada multi-meter sebenarnya, maka pada multimeter pada Livewire juga memiliki mode-mode pengukuran untuk mengukur arus (A), tegangan DC (V) dan tegangan AC (RMS).

Disamping komponen ***Analogue Multimeter*** juga terdapat komponen ***Digital Multimeter*** yang terkelompok pada sub-***Measuring.*** Tampilan dari komponen digital multimeter memberikan tampilan instrumen pengukuran yang lebih modern seperti pada gambar di bawah ini.



Alat ukur penting yang sering kita gunakan selanjutnya pada simulasi rangkaian menggunakan Livewire adalah *Oscilloscope*. Oscilloscope merupakan instrumen yang digunakan untuk mengukur dan merekam sinyal yang terdapat pada suatu rangkaian.

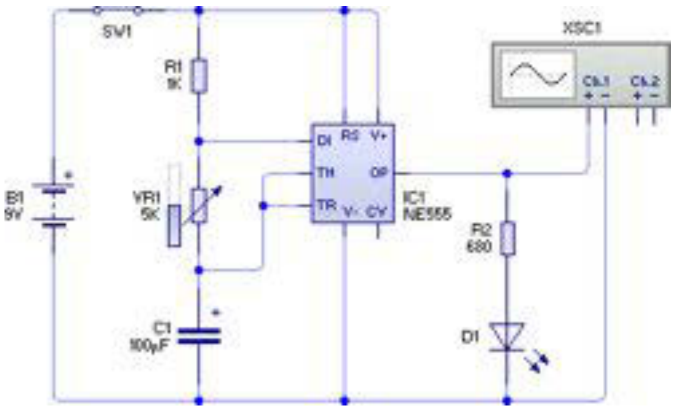
Untuk menambahkan komponen Oscilloscope pada rangkaian Timer IC 555, maka terlebih dahulu kita hapus terlebih dahulu multimeter yang terdapat seperti pada gambar di atas. Untuk menghapus cukup arahkan kursor ke komponen multimeter, kemudian klik satu kali komponen tersebut dan selanjutnya tekan tombol “DEL” pada keyboard laptop/PC kalian.

Langkah selanjutnya adalah menambahkan komponen *Oscilloscope* pada halaman kerja. Komponen ini terletak pada kelompok Measuring. Sebuah *oscilloscope* dapat mengukur dua buah sinyal pada saat bersamaan. Setiap sinyal di ukur melalui sebuah chanel yang memiliki pin positif (+) dan pin

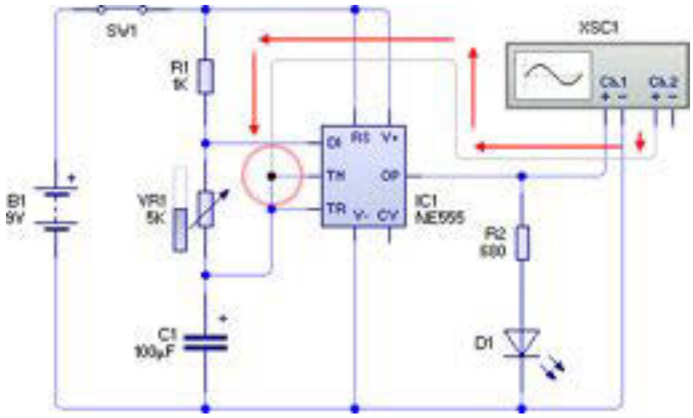


negatif (-).*Oscilloscope* akan mencatat nilai beda potensial dari kedua pin tersebut pada setiap chanel.

Hubungkan pin positif (+) chanel 1 di antara output IC 555 dengan resistor R2 680Ω. Kemudian hubungkan pin negatif (-) chanel 1 *oscilloscope* ke ground dari rangkaian. Pengukuran ini akan mengukur sinyal output yang dikeluarkan oleh IC timer 555.



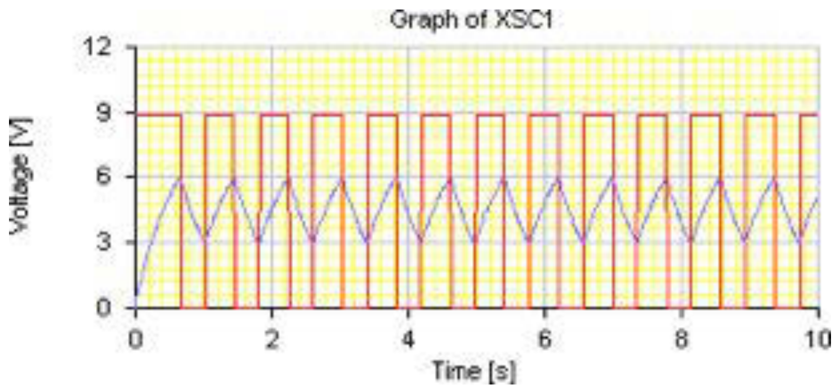
Selanjutnya chanel 2 dari *oscilloscope* akan digunakan untuk mengukur sinyal trigger yang dihasilkan oleh kapasitor C1. Untuk itu hubungkan pin positif (+) chanel 2 ke titik percabangan TH pada *oscilloscope* seperti tampak pada gambar dibawah ini. Berikutnya adalah menghubungkan pin negatif (-) chanel 2 dari *oscilloscope* ke titik ground.





Tahap akhir dari penggunaan *oscilloscope* adalah menyiapkan grafis untuk membaca dan menampilkan hasil pengukuran *oscilloscope* tersebut. Untuk itu, klik kanan mouse pada komponen oscilloscope **XSC1** dan pilih menu ***Add Graph*** dari pop-up menu yang ditampilkan. Kemudian tekan dan tahan tombol mouse sebelah kiri titik (a), lalu tarik mouse ke arah kanan bawah titik (b) seperti pada gambar.

Tekan tombol RUN untuk menjalankan kembali simulasi Livewire dan perhatikan perubahan sinyal yang dihasilkan oleh IC Timer 555 akan ditampilkan pada grafis yang telah kita siapkan tadi.



Dari hasil simulasi Lifewire, tampak dua buah hasil sinyal output yang ditampilkan seperti pada gambar di atas. Sinyal chanel 1 ditampilkan dengan warna merah dan membentuk sinyal kotak, sedangkan sinyal chanel 2 ditampilkan dengan warna biru dan membentuk sinyal segi tiga.

Dari gambar pengukuran simulasi oscilloscope tersebut dapat kita analisa, output sinyal chanel 1 berubah secara konstan digital dari 0 volt menuju 9 volt bolak-balik secara periodik. Sedangkan sinyal chanel 2 bergerak secara bertahap dari 3 volt menuju 6 volt bolak-balik secara periodik. Hal ini mengindikasikan proses yang terjadi pada kapasitor C1 ketika melalui proses pengisian dan pengosongan.



Rangkuman

Program aplikasi *Livewire-Professional Edition* versi 1.11 merupakan software yang digunakan untuk melakukan simulasi rangkaian elektronik dan listrik. Penggunaan program *Livewire-Professional Edition* versi 1.11 relatif sangat mudah dipahami oleh siswa pemula karena tampilan visual yang menarik dan fasilitas fitur yang cukup lengkap sehingga memudahkan dalam mempelajari dan menganalisa suatu rangkaian.

Komponen elektronika dan listrik pada livewire dikelompokkan berdasarkan fungsi dari komponen tersebut. Berikut garis besar pengelompokan komponen pada livewire ;

- Power Supplies
- Connector
- Input Components
- Passive Components
- Discretes Semiconductors
- Logic Gates
- Integrated Circuits
- Output Components
- Measuring

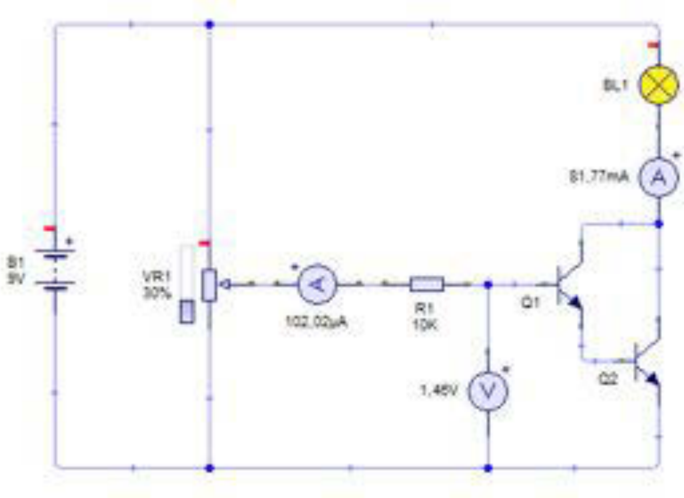


Tugas

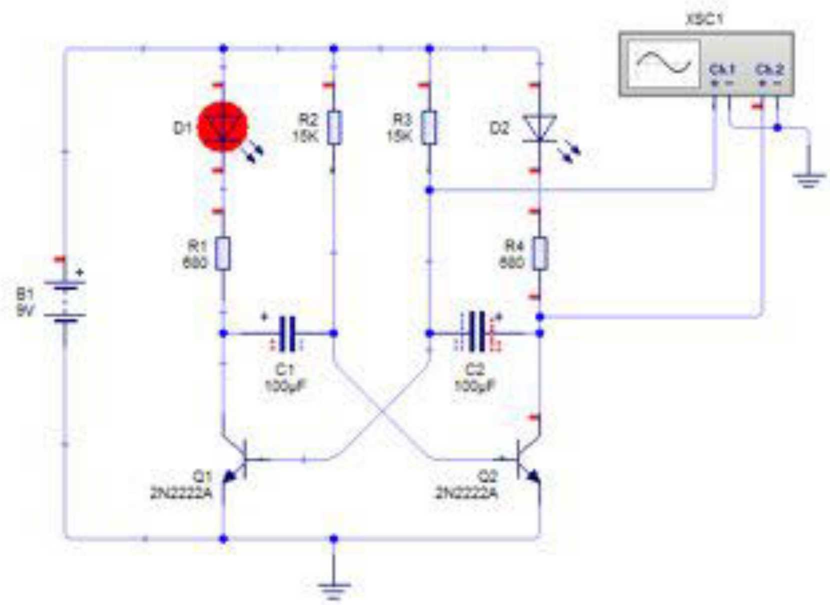
Perhatikan rangkaian IC timer 555 astable pada pembelajaran sebelumnya, ubahlah nilai kapasitor C1 100µF menjadi 10µF! Perhatikan apa yang terjadi dengan output IC 555 dan hitunglah serta perbandingkan nilai periode (1/f) antara hitungan dan pembacaan pada oscilloscope!

Tes Formatif

1. Buatlah rangkaian transistor darlington seperti gambar di bawah ini dan jelaskan prinsip kerja rangkaian tersebut !



2. Buatlah rangkaian transistor berosilasi seperti gambar rangkaian dibawah ini!





KEGIATAN BELAJAR 4

Sebelum proses pembelajaran di kelas berlangsung, sebaiknya siswa mempersiapkan diri dengan belajar mandiri sesuai dengan urutan materi yang akan diberikan.Sebagai gambaran kegiatan belajar siswa seperti pada tabel berikut :

NO	KEGIATAN SISWA	KETERANGAN
1	Persiapan Kegiatan 1 1. Siswa membaca materi pendahuluan 2. Siswa mempelajari materi identifikasi arsitektur Mikrokontroller 3. Siswa mempelajari hardware Mikrokontroller dengan latihan menggambar pengawatan Mikrokontroller 4. Siswa mencoba mengerjakan soal tes formatif 1	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 1
2	Persiapan Kegiatan 2 1. Siswa membaca materi pengertian program 2. Siswa mempelajari materi teknik pemrograman Mikrokontroller 3. Siswa mempelajari instruksi sederhana pada program Mikrokontroller 4. Siswa mencoba mengerjakan soal tes formatif 2	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 2
3	Persiapan Kegiatan 3 1. Siswa mempelajari materi transfer program ke dalam Mikrokontroller dengan menggunakan software BASCOM-programmer. 2. Siswa mempelajari pembuatan ladder diagram 3. Siswa mencoba mengerjakan soal tes formatif 3	Kegiatan ini pada prinsipnya siswa belajar mandiri untuk menerima materi dari guru sesuai kegiatan

Selanjutnya siswa mendengarkan penyampaian materi pembelajaran di setiap pertemuan oleh guru serta menyesuaikan dengan model pembelajran yang



digunakan. Misalnya saatnya harus aktif mengerjakan soal maupun praktikum, maka siswa juga harus aktif dan kreatif. Melalui langkah kegiatan pembelajaran yang saling melengkapi diharapkan siswa dapat mencapai kompetensi yang distandarkan.

A. Tujuan Pembelajaran

Setelah mempelajari materi tentang arsitektur Mikrokontroller, diharapkan siswa dapat:

1. mengidentifikasi arsitektur Mikrokontroller
2. mengidentifikasi blok dalam Mikrokontroller

B. Uraian Materi

- Dasar sistem kendali Mikrokontroller, komponen dan spesifikasinya serta perbandingan sistem kendali Mikrokontroller dengan sistem kendali yang lain.
- Teknik pemrograman Mikrokontroller.
- Teknik pemasangan dan pengawatan peralatan input output.
- Penggunaan alat pemrogram dengan komputer yang dilengkapi dengan software ladder
- Pengoperasian sistem kendali Mikrokontroller

C. Alokasi Waktu

4 jam pelajaran

D. Metode Pembelajaran

Teori dan Praktek

E. Media pembelajaran

- PC/Notebook
- Windows 7
- BASCOM-Prgogrammer
- Minimum System AVR mikrokontroller

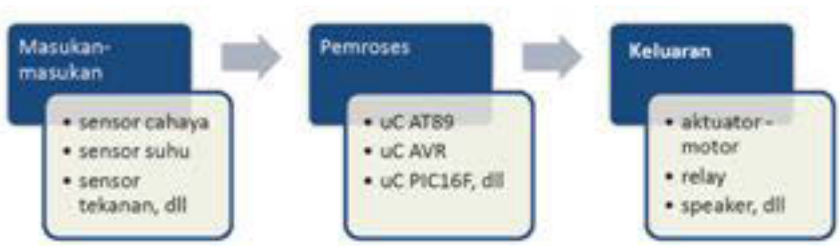


KEGIATAN 1

4.1 Mengenal Mikrokontroller

Jika kita bicara tentang Mikrokontroler, maka tidak terlepas dari pengertian atau definisi tentang Komputer itu sendiri, mengapa? Ada kesamaan-kesamaan antara Mikrokontroler dengan Komputer (atau Mikrokomputer), antara lain:

- Sama-sama memiliki unit pengolah pusat atau yang lebih dikenal dengan **CPU** (***Central Processing Unit***);
- **CPU** tersebut sama-sama menjalankan program dari suatu lokasi atau tempat, biasanya dari **ROM** (*Read Only Memory*)1 atau **RAM** (*Random Access Memory*)2;
- Sama-sama memiliki **RAM** yang digunakan untuk menyimpan data-data sementara atau yang lebih dikenal dengan variabel-variabel;
- Sama-sama memiliki beberapa keluaran dan masukan (**I/O**) yang digunakan untuk melakukan komunikasi timbal-balik dengan dunia luar, melalui sensor (masukan) dan aktuator (keluaran), perhatikan bagan yang ditunjukkan pada Gambar 1.1.



Gambar 4.1. Bagan masukan, pemrosesan hingga keluaran

Lantas apa yang membedakan antara Mikrokontroler dengan Komputer atau Mikrokomputer?

Mikrokontroler adalah versi mini dan untuk aplikasi khusus dari Mikrokomputer atau Komputer!

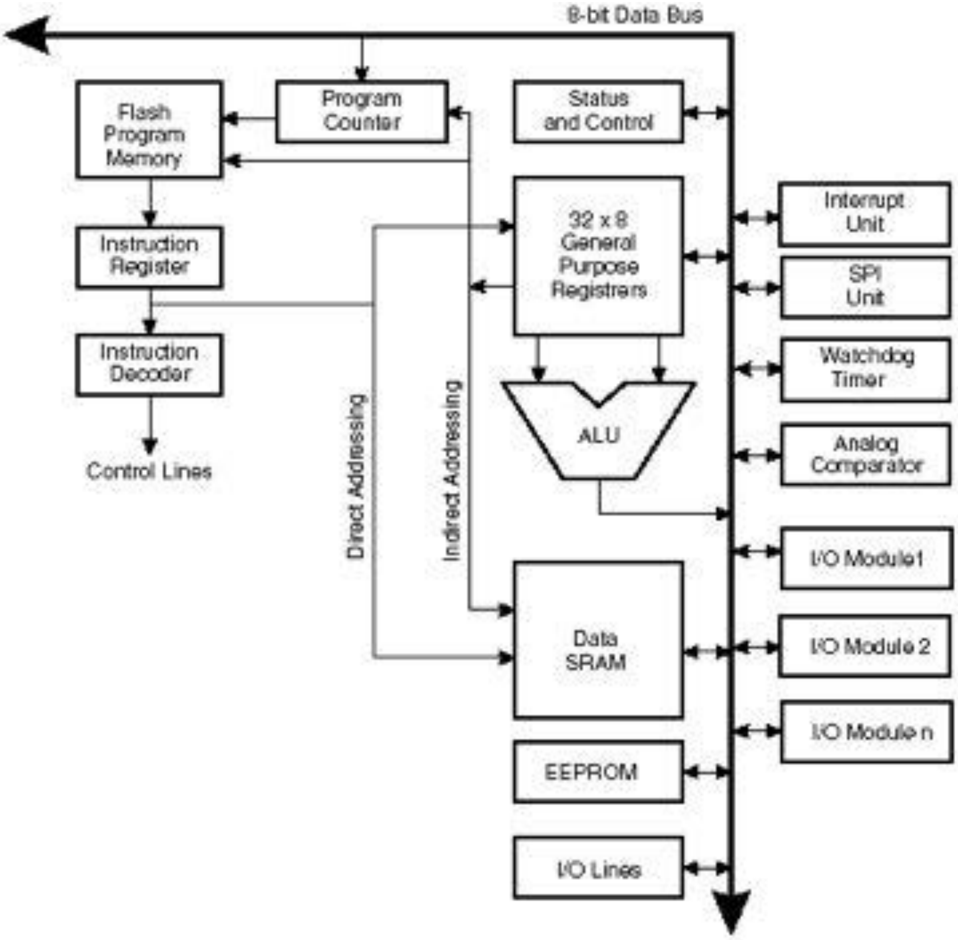


Berikut daftar kesamaan yang pernah kemukakan sebelumnya dengan menekankan pada perbedaan antara Mikrokontroler dan Mikrokomputer:

- **CPU** pada sebuah Komputer berada eksternal dalam suatu sistem, sampai saat ini kecepatan operasionalnya sudah mencapai lebih dari 2,5 GHz, sedangkan CPU pada Mikrokontroler berada didalam (internal) sebuah *chip*, kecepatan kerja atau operasionalnya masih cukup rendah, dalam orde MHz (misalnya, 24 MHz, 40 MHz dan lain sebagainya). Kecepatan yang relatif rendah ini sudah mencukupi untuk aplikasi-aplikasi berbasis mikrokontroler.
- Jika ***CPU***pada mikrokomputer menjalankan program dalam ***ROM***atau yang lebih dikenal dengan ***BIOS***(***Basic I/O System***) pada saat awal dihidupkan, kemudian mengambil atau menjalankan program yang tersimpan dalam hard disk. Sedangkan mikrokontroler sejak awal menjalankan program yang tersimpan dalam ***ROM***internal-nya (bisa berupa ***Mask ROM***atau ***Flash PEROM***atau ***Flash ROM***). Sifat memori program dalam mikrokontroler ini *non-volatile*, artinya tetap akan tersimpan walaupun tidak diberi catu daya.
- **RAM** pada mikrokomputer bisa mencapai ukuran sekian GByte dan bisa di-upgrade ke ukuran yang lebih besar dan berlokasi di luar **CPU**-nya, sedangkan RAM pada mikrokontroler ada di dalam *chip* dan kapasitasnya rendah, misalnya 128 byte, 256 byte dan seterusnya dan ukuran yang relatif kecil inipun dirasa cukup untuk aplikasi-aplikasi mikrokontroler.
- keluaran dan masukan (**I/O**) pada mikrokomputer jauh lebih kompleks dibandingkan dengan mikrokontroler, yang jauh lebih sederhana, selain itu, pada mikrokontroler akses keluaran dan masukan bisa per bit.
- Jika diamati lebih lanjut, bisa dikatakan bahwa Mikrokomputer atau Komputer merupakan komputer serbaguna atau *general purpose computer*, bisa dimanfaatkan untuk berbagai macam aplikasi (atau perangkat lunak). Sedangkan mikrokontroler adalah *special purpose computer* atau komputer untuk tujuan khusus, hanya satu macam aplikasi saja.



Perhatikan Gambar 4.2, agar Anda mendapatkan gambaran tentang mikrokontroler lebih jelas.



Gambar 4.2. Diagram Blok mikrokontroler

ALU, ***Instruction Decoder***, ***Accumulator*** **dan** ***Control Logic***, sebagaimana ditunjukkan pada Gambar 1.2, merupakan "**Otak**" dari mikrokontroler. Jantungnya berasal dari detak kristal oscilator. Disamping itu juga terdapat berbagai macam periferal seperti **SFR** (***Special Function Register***) yang bertugas menyimpan data-data sementara selama proses berlangsung. ***Instruction Decoder*** bertugas menerjemahkan setiap instruksi yang ada di dalam **Program Memory** (hasil dari pemrograman yang di buat sebelumnya). Hasil penerjemahan tersebut merupakan suatu operasi yang harus dikerjakan oleh **ALU** (***Arithmetic Logic Unit***).



Accumulator yang kemudian menghasilkan sinyal-sinyal kontrol ke seluruh periferal yang terkait melalui **Control Logic**. **Memori RAM** atau *RAM Memory* bisa digunakan sebagai tempat penyimpan sementara, sedangkan **SFR** (***Special Function Register***) sebagian ada yang langsung berhubungan dengan **I/O** dari mikrokontroler yang bersangkutan dan sebagian lain berhubungan dengan berbagai macam operasional mikrokontroler.

ADC atau ***Analog to Digital Converter*** digunakan untuk mengubah data-data analog menjadi digital untuk diolah atau diproses lebih lanjut.

Timer atau ***Counter*** digunakan sebagai pewaktu atau pencacah. Sebagai pewaktu fungsinya seperti sebuah jam digital dan bisa diatur cara kerjanya. Sedangkan pencacah lebih digunakan sebagai penghitung atau pencacah *event* atau bisa juga digunakan untuk menghitung berapa jumlah pulsa dalam satu detik dan lain sebagainya. Biasanya sebuah mikrokontroler bisa memiliki lebih dari 1 *timer*.

EEPROM berfungsi sama seperti RAM hanya saja tetap akan menyimpan data walaupun tidak mendapatkan sumber listrik/daya.Sedangakan port-port I/O untuk masukan/keluaran, untuk melakukan komunikasi dengan periferal eksternal mikrokontroler seperti sensor dan aktuator.

Beberapa catatan mikrokontroler lainnya adalah:

- ‘Tertanam’ (atau *embedded*) dalam beberapa piranti atau dikenal dengan istilah *embedded system* atau *embedded controller*;
- Terdedikasi untuk satu macam aplikasi saja.
- Hanya membutuhkan daya yang (cukup) rendah (*low power*) sekitar 50 mWatt (Anda bandingkan dengan komputer yang bisa mencapai 50 Watt lebih);
- Memiliki beberapa keluaran maupun masukan yang terdedikasi, untuk tujuan atau fungsi-fungsi khusus;
- Kecil dan relatif lebih murah (seri AT89 di pasaran serendah-rendahnya bisa mencapai Rp 15.000,00, mikrokontroler AVR di pasaran saat ini juga relatif murah sedangkan Arduino bisa mencapai Rp. 200.000,-);



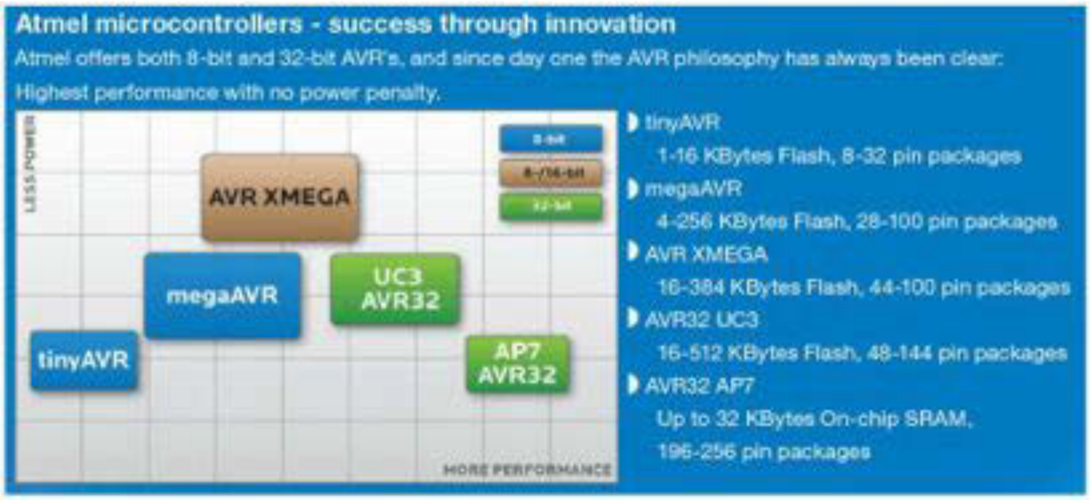
- Seringkali tahan-banting, terutama untuk aplikasi-aplikasi yang berhubungan dengan mesin atau otomotif atau militer.
- Mikrokontroler yang beredar saat ini dibedakan menjadi dua macam, berdasarkanarsitekturnya:
  - Tipe **CISC** atau ***Complex Instruction Set Computing*** yang lebih kaya instruksi tetapi fasilitas internal secukupnya saja (seri AT89 memiliki 255 instruksi);
  - Tipe **RISC** atau ***Reduced Instruction Set Computing*** yang justru lebih kaya fasilitas internalnya tetapi jumlah instruksi secukupnya (seri PIC16F hanya ada sekitar 30-an instruksi).

4.2. Pengetahuan Dasar Mikrokontroler AVR

Keluarga Mikrokontroler AVR merupakan mikrokontroler dengan arsitektur modern. Perhatikan Gambar 1.3, Atmel membuat 5 (lima) macam atau jenis mikrokontroler AVR, yaitu:

- **TinyAVR** Mikrokontroler mini (hanya 8 sampai 32 pin) serbaguna dengan Memori Flash untuk menyimpan program hingga 16K Bytes, dilengkapi SRAM dan EEPROM 512 Bytes.
- **MegaAVR** Mikrokontroler dengan unjuk-kerja tinggi, dilengkapi pengali perangkat keras (*HardwareMultiplier*), mampu menyimpan program hingga 256 KBytes, dilengkapi EEPROM 4K Bytes dan SRAM 8K Bytes.
- **AVR XMEGA** Mikrokontroler AVR 8/16-bit XMEGA memiliki periferal baru dan canggih dengan unjukkerja, sistem *Event* dan DMA yang ditingkatkan, serta merupakan pengembangan keluarga AVR untuk pasar *low power* dan *high performance* (daya rendah dan unjuk-kerja tinggi).
- **AVR32 UC3** Unjuk-kerja tinggi, mikrokontroler flash AVR32 32-bit daya rendah. Memiliki flash hingga 512 KByte dan SRAM 128 KByte.
- **AVR32 AP7** Unjuk-kerja tinggi, prosesor aplikasi AVR32 32-bit daya rendah, memiliki SRAM hingga 32 KByte.





Gambar 4.3. Jenis Mikrokontroler Atmel

Di Indonesia jenis mikrokontroller yang paling populer adalah **tinyAVR** dan **megaAVR**. Perbedaan jenis-jenis tersebut terletak dari fasilitas, atau lebih dikenal dengan fitur-fiturnya. Jenis **TinyAVR** merupakan mikrokontroler dengan jumlah pin yang terbatas dan memiliki fitur-fitur yang terbatas dibandingkan yang **megaAVR**. Semua mikrokontroler AVR memiliki set instruksi (*assembly*) dan organisasi memori yang sama, dengan demikian berpindah-pindah (walaupun tidak disarankan) antar mikrokontroler AVR menjadi hal yang tidak sulit atau bisa dikatakan cukup mudah.

4.3 Arsitektur Mikrokontroller Atmega16

AVR merupakan kependekan dari Alf (Egil Bogen) and Vegard (Wollan) 's Risc processor. Mikrokontroller Atmega16 adalah mikrokontroller AVR 8 bit buatan ATMEL yang memiliki arsitektur RISC (Reduce Instruction Set Computing). Instruksi dikemas dalam kode 16 bit dan dijalankan hanya dengan satu siklus clock. Struktur I/O yang baik dengan sedikit komponen tambahan diluar. Fasilitas internal yang terdapat pada mikrokontroller ini adalah oscillators, timers, UART, SPI, pull-up resistors, pulse width modulation (PWM), ADC, analog comparator dan watch-dog timers.



4.3.1 Fitur

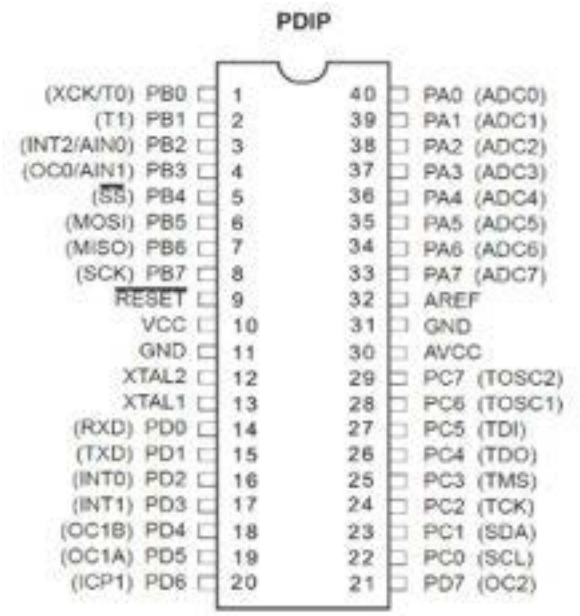
- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 130 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
  - 8K Bytes of In-System Self-Programmable Flash
  - Endurance: 10,000 Write/Erase Cycles
  - In-System Programming by On-chip Boot Program
  - True Read-While-Write Operation
  - 512 Bytes EEPROM, Endurance: 100,000 Write/Erase Cycles
  - 512 Bytes Internal SRAM
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
  - Byte-oriented Two-wire Serial Interface
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad QFN/MLF
- Operating Voltages
  - 2.7 - 5.5V for Atmega16L
  - 4.5 - 5.5V for Atmega16



Keunggulan AVR adalah pengkombinasian banyak instruksi dengan 32 general purpose working registers. Kesemua 32 register tersambung langsung ke Arithmetic Logic Unit (ALU), Mengijinkan dua register bebas dapat diakses dengan intruksi yang dieksekusi hanya dengan satu siklus clock. Hasilnya adalah suatu arsitektur dengan kode yang lebih efisien dan 10 kali lebih cepat dibandingkan dengan mikrokontroller konvensional CISC (Complex Instruction set Computing).

Mikrokontroller ini dibuat dengan Atmel's high density nonvolatile memory technology. Program memory ISP Flash pada chip tunggal ini dapat diprogram dalam mode In-System melalui saluran interface serial SPI.

4.3.2 Konfigurasi Pin



Gambar 4.4. Diagram Pin Mikrokontroler AVR ATMega16 tipe DIP



4.3.3 Deskripsi Pin

Vcc Power Supply

GND Ground

Port A (PA7..PA0) Port A berfungsi sebagai masukan analog ke A/D Converter. Selain itu Port A juga berfungsi sebagai port I/O dua arah 8 bit apabila ADC tidak digunakan. Pin pada port dilengkapi pula dengan resistor pull-up internal yang dapat diaktifkan untuk setiap bit yang dipilih.

Port Pin	Fungsi Alternatif
PA0	ADC0 (ADC input channel 0)
PA1	ADC1 (ADC input channel 1)
PA2	ADC2 (ADC input channel 2)
PA3	ADC3 (ADC input channel 3)
PA4	ADC4 (ADC input channel 4)
PA5	ADC5 (ADC input channel 5)
PA6	ADC6 (ADC input channel 6)
PA7	ADC7 (ADC input channel 7)

Port B (PB7..PB0) Port B berfungsi sebagai port I/O dua arah 8 bit dengan resistor pull-up internal yang dapat diaktifkan untuk setiap bit yang dipilih.

Port Pin	Fungsi Alternatif
PB0	T0 (Timer/Counter0 External Counter Input) XCK (USART External Clock Input/Output)
PB1	T1 (Timer/Counter1 External Counter Input)
PB2	AIN0 (Analog Comparator Positive Input) INT2 (External Interrupt 2 Input)
PB3	AIN1 (Analog Comparator Negative Input) OC0 (Timer/Counter0 Output Compare Match Output)
PB4	$\overline{SS}$ (SPI Slave Select Input)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB7	SCK (SPI Bus Serial Clock)



Port C (PC7..PC0) Port C berfungsi sebagai port I/O dua arah 8 bit dengan resistor pull-up internal yang dapat diaktifkan untuk setiap bit yang dipilih.

Port Pin	Fungsi Alternatif
PC0	SCL (Two-wire Serial Bus Clock Line)
PC1	SDA (Two-wire Serial Bus Data Input/Output Line)
PC6	TOSC1 (Timer Oscillator Pin 1)
PC7	TOSC2 (Timer Oscillator Pin 2)

Port D (PD7..PD0) Port D berfungsi sebagai port I/O dua arah 8 bit dengan resistor pull-up internal yang dapat diaktifkan untuk setiap bit yang dipilih.

pull-up internal yang dapat diaktifkan untuk setiap bit yang dipilih.

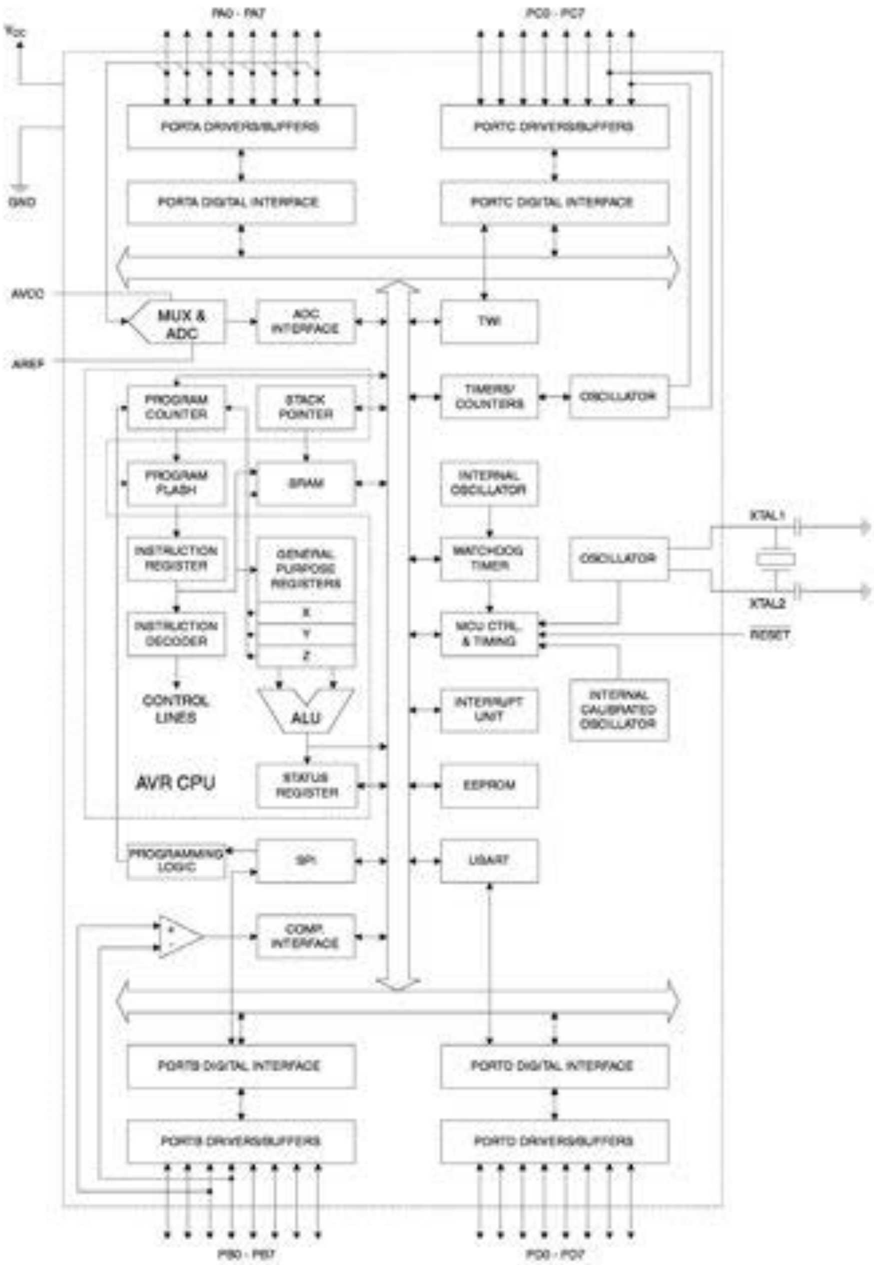
Port Pin	Fungsi Alternatif
PD0	RXD (USART Input Pin)
PD1	TXD (USART Output Pin)
PD2	INT0 (External Interrupt 0 Input)
PD3	INT1 (External Interrupt 1 Input)
PD4	OC1B (Timer/Counter1 Output Compare B Match Output)
PD5	OC1A (Timer/Counter1 Output Compare A Match Output)
PD6	ICP1 (Timer/Counter1 Input Capture Pin)
PD7	OC2 (Timer/Counter2 Output Compare Match Output)

RESET Pin ini berfungsi untuk mereset mikrokontroller. Aktif low. Artinya jika pin ini diberi input logika 0, maka mikrokontroller akan ter-reset.

XTAL1 Pin ini tersambung ke kristal eksternal. Dilihat dari rangkaian internal, pin ini sebagai input ke inverting Oscillator amplifier dan input ke rangkaian operasinal clock internal.

XTAL2 Pin ini tersambung ke kristal eksternal. Merupakan keluaran dari inverting Oscillator amplifier

AVCC	Berfungsi sebagai input power supply untuk Port A dan A/D Converter. Pin ini harus disambung pada Vcc dan sebaiknya dipasangi low pass filter. Jika tidak tersambung ke Vcc maka A/D Converter tidak berfungsi.
AREF	Adalah pin tegangan reference analog untuk A/D Converter.



Gambar 4.6 Digram Blok Mikrokontroller Atmega16

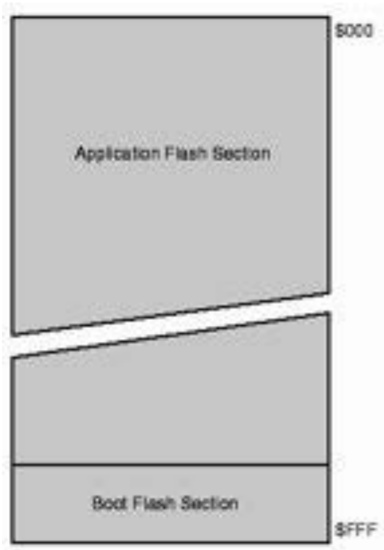


4.4 AVR Atmega16 Memory

In-System Reprogrammable Flash Program Memory

Mikrokontroler Atmega16 memiliki 16 Kbytes On-chip In-System Reprogrammable Flash memory untuk meyimpan program. Karena semua instruksi AVR lebarnya 16 atau 32 bit, maka memory flash diorganisasi sebagai 4K x 16. Untuk keamanan software, Flash memory space dibagi dalam dua seksi, yaitu : *Boot Program section* dan *Application Program section*.

Flash memory meiliki ketahanan paling sedikit 10,000 kali siklus *write/erase*. Atmega16 Program Counter (PC) lebarnya adalah 12 bits yang dapat mengalamati program sebesar 4K lokasi memory.

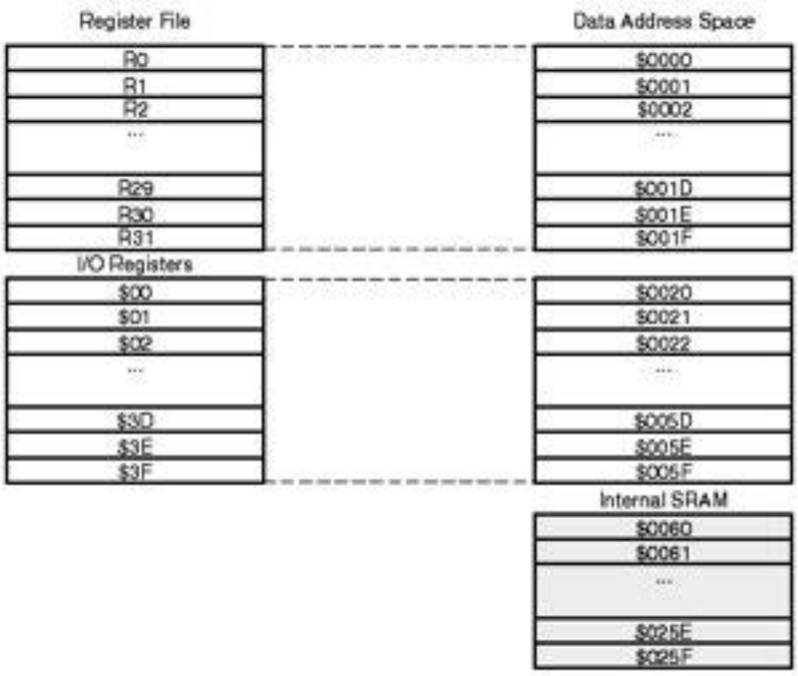


Gambar 4.7 Peta Memory Program



SRAM Data Memory

608 lokasi alamat data memory adalah Register File, I/O Memory dan internal data SRAM. 96 lokasi alamat pertama adalah Register File dan I/O Memory. 512 lokasi alamat berikutnya adalah internal data SRAM.



Gambar 4.8 Peta Memory Data

Pengalamatan langsung dapat mencapai semua *space memory* data. Pengalamatan tidak langsung dengan mode *displacement* hanya dapat mencapai 63 lokasi memory dari alamat dasar yang diberikan oleh register Y atau Z.

Ketika menggunakan register mode pengalamatan tidak langsung dengan pre-*decrement* dan *post-increment* aotomatis, isi register X, Y dan Z akan di-decrement atau di-increment.

Dengan mode pengalamatan tersebut dapat mengakses 32 general purpose working registers, 64 I/O Registers dan the 512 bytes of internal data SRAM.



EEPROM Data Memory

The Atmega16 memiliki 512 bytes memory data EEPROM yang tahan paling sedikit 100.000 kali siklus *write/erase***.** Ketika EEPROM sedang dibaca, CPU akan berhenti bekerja selama empat siklus clock sebelum instruksi berikutnya dieksekusi. Dan pada saat EEPROM sedang ditulisi CPU akan berhententi selama dua siklus clock.

Untuk menulis EEPROM diperlukan waktu programming selama 8,4 ms

Register yang berhubungan dengan EEPROM adalah sebagai berikut :

EEPROM Address Register – EEARH dan EEARL

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	–	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	X	
	X	X	X	X	X	X	X	X	

Gambar 4.9 Peta Register EERH dan EERL

I/O Memory

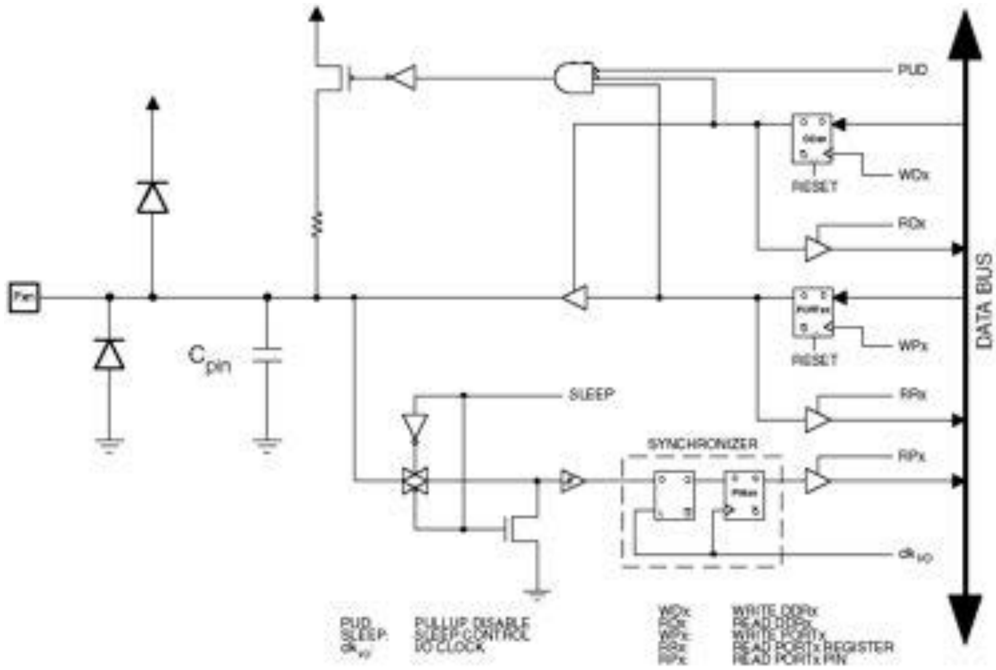
Semua I/O dan peripheral Atmega16 ditempatkan pada I/O Space. Untuk mengakases lokasi I/O menggunakan instruksi IN dan OUT yang dapat menstransfer data antara 32 general purpose working registers dan I/O space. Register I/O yang berada pada alamat antara 0x00 sampai dengan 0x1F dapat diakses langsung per bit denganmenggunakan instruksi SBI dan CBI. Pada register ini, nilai bit tunggal dapat dicek menggunakan instruksi SBIS dan SBIC.

Ketika menggunakan instruksi khusus untuk I/O IN dan OUT, alamat I/O 0x00 – 0x3F harus digunakan. Ketika pengalamatan register I/O sebagai data space menggunakan instruksi LD dan ST, 0x20 harus ditambahkan pada alamat ini.



4.5 I/O Ports

Semua port AVR memiliki fungsi *true Read-Modify-Write* ketika digunakan sebagai port I/O digital.Maksudnya bahwa arah satu port pin dapat dirubah tanpa merubah arah port pin yang lain dengan instruksi SBI dan CBI. Hal yang sama juga berlaku untuk mengubah nilai keluaran (jika dikonfigurasi sebagai output) atau *enable/disable* resistor *pull-up* internal jika dikonfigurasi sebagai input. Setiap buffer keluaran memiliki karakteristik pengendalian yang simetri baik untuk *high sink* maupun *source*. *Driver* pin cukup kuat untuk men-*drive* LED secara langsung. Semua port memeiliki selektor resistor *pull-up* sendiri-sendiri dengan suatu sumber tegangan dengan resistansi yang bervariasi. Semua pin I/O memiliki dioda proteksi ke Vcc dan Gorund.



Gambar 4.10 Rangkaian equivalent pin I/O

Pada penjelasan berikutnya dipakai symbol “x” untuk mewakili port dan symbol “n” unutk mewakili bit, contoh PORTB3 untuk port B bit 3 dapat ditulis sebagai PORTxn.

Tiga buah lokasi memory I/O terletak pada setiap port. Satu untuk register data – PORTx, Data Direction Register – DDRx dan Port Input Pins – PINx. Lokasi Port Port Input Pins I/O adalah read only ketika Register Data dan Data Direction



Register sedang dibaca atau ditulisi. Pull-up Disable – PUD bit dalam SFIOR memungkinkan fungsi pull-up untuk semua pin jika diset.

Setiap pin port terdiri dari tiga register bits, yaitu DDxn, PORTxn dan PINxn. DDxn bits diakses pada alamat DDRx I/O, PORTxn bits pada PORTx I/O address dan PINxn bits pada PINx I/O.

DDxn bit pada Register DDRx dipergunakan untuk mengatur arah data pin. Jika DDxn ditulisi dengan logika 1, maka Pxn dikonfigurasi sebagai pin keluaran. Jika DDxn diberilogika 0, maka Pxn dikonfigurasi sebagai pin masukan.

Jika PORTxn diberi logika 1 ketika pin dikonfigurasi sebagai input, resistor pull-up akan diaktifkan. Untuk melepas saklar resistor *pull-up*, PORTxn harus diberi logika 0 atau pin dikonfigurasi sebagai pin keluaran. Ketika pin reset aktif, pin port akan dikodisikan *tri-stated* meskipun tidak ada pulsa *clock*.

Jika PORTxn diberi logika 1 ketika pin dikonfigurasi sebagai pin keluaran, maka pin port akan di-drive ke logika tinggi (1). Jika PORTxn diberi logika 0 ketika pin dikonfigurasi sebagai pin keluaran, maka pin port akan di-drive ke logika rendah (0).

Ketika *switching* antara *tri-state* ({DDxn, PORTxn} = 0b00) and output *high* ({DDxn, PORTxn} = 0b11), dimungkinkan sebuah *intermediate state* dengan *pull-up* ({DDxn, PORTxn} =0b01) atau output *low* ({DDxn, PORTxn} = 0b10) akan muncul. Pada umumnya *pull-up enabled state* dapat diakses penuh sebagai *high-impedant environment* tanpa perbedaan antara *strong high driver* dan *pull-up*. Jika tidak demikian, maka PUD bit dalam register SFIOR dapat diset ke disable untuk semua *pull-up* pada semua port.

Switching antara input dengan *pull-up* dan output *low* menghasilkan masalah yang sama. Pemakai harus menggunakan *tri-state* ({DDxn, PORTxn} = 0b00) atau output *high state*({DDxn, PORTxn} = 0b10) sebagai suatu *intermediate step.*

Tabel 1.1 Konfigurasi Pin Port

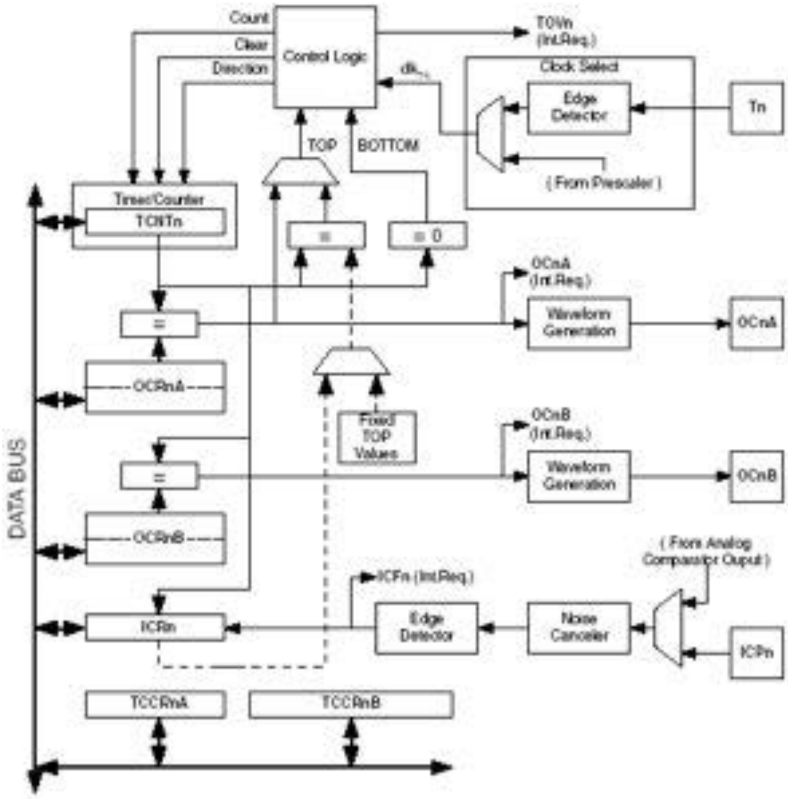


DDxn	PORTxn	PUD	I/O	Pull-up	Keterangan
0	0	x	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn akan menjadi sumber arus, jika external pulled low
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	x	Output	No	Output Low (Sink)
1	1	x	Output	No	Output High (Source)

4.6 Timer/Counter

Mikrokontroller Atmega16 memiliki tiga buah Timer/Counter yaitu Timer/Counter0 8-bit dengan PWM, Timer/Counter1 16-bit dan Timer/Counter2 8-bit dengan PWM.

Timer/Counter1 16-bit



Gambar 4.12 Diagram Block Timer/Counter1 16 bit

Timer/Counter (TCNT1), *Output Compare Registers* (OCR1A/B) dan Input *Capture Register* (ICR1) semuanya adalah register 16-bit. Prosedur khusus harus diikuti untuk mengakses register 16-bit.



Timer/Counter Control Registers (TCCR1A/B) adalah register 8-bit dan tidak memiliki akses ke CPU. Sinyal *Interrupt requests* (dalam gambar disingkat menjadi Int.Req.) dapat dilihat pada *Timer Interrupt Flag Register* (TIFR).
Semua *interrupts di-masker* secara individual dengan *Timer Interrupt Mask Register* (TIMSK). TIFR and TIMSK tidak ditunjukkan dalam gambar dan juga dipergunakan oleh timer yang lain. The Timer/Counter dapat di-clock menggunakan clock internal, melalui prescaler, atau dapat juga di-clock dari sumber eksternal melalui pin T1.

Block Clock Select logic mengatur sumber dan edge clock. the Timer/Counter menggunakan pulsa clock ini untuk meng-*increment* (atau *decrement*) nilai counter. Timer/Counter tidak aktif apabila tidak ada sumber clock yang dipilih. Keluaran dari Clock Select logic adalah sesuai dengan timer clock (clkT1).

Setiap saat nilai *Double buffered Output Compare Register* (OCR1A/B) dibandingkan dengan nilai Timer/Counter. Hasil perbandingan dapat dipergunakan oleh *Waveform Generator* untuk menghasilkan keluaran PWM atau *variable frequency* pada pin Output Compare (OC1A/B). *Compare Match event* juga men-*set Compare Flag* (OCF1A/B)
yang juga dapat dipakai untuk meng-*generate* output *compare interrupt request*.

Pemakaian Timer/Counter1 dapat digunakan sebagai tunda waktu (delay), pada contoh dibawah ini bahwa kristal yang dipergunakan modul mikrokontroller adalah 12 MHz. Dengan kristal sebesar ini, maka satu detik sama dengan 12.000.000 pulsa clock. Berikut ini algoritma membuat tunda waktu 1 detik.

1. Mengaktifkan interrupt TOIE1 dengan cara men-set bit2 – TOIE1: Timer/Counter1, Overflow Interrupt Enable pada register Timer/Counter Interrupt Mask – TIMSK

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Jika bit ini diberi logika 1, maka Timer/*Counter1 overflow interrupt* di-*enable*-kan dan I-flag di Status Register akan di-set 1 *(interrupts globally enabled*)



2. Menentukan nilai pembagi prescaler dengan men-set register Timer/Counter1 Control Register B – TCCR1B

Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit ke 0 sampai bit ke 2 [CS12,CS11,CS10] digunakan untuk memilih sumber clock yang akan digunakan Timer/Counter, dengan keterangan sebagai berikut :

CS12	CS11	CS10	Keterangan
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	clkI/O/1 (No prescaling)
0	1	0	clkI/O/8 (From prescaler)
0	1	1	clkI/O/64 (From prescaler)
1	0	0	clkI/O/256 (From prescaler)
1	0	1	clkI/O/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

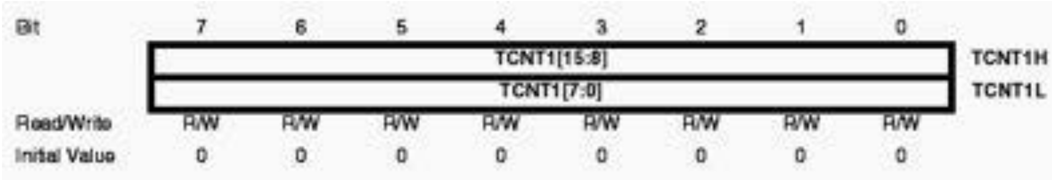
Dengan memilih sumber clock, dihitung nilai yang paling baik dengan selisih yang paling kecil. Apabila dihitung secara manual akan diperoleh hasil perbandingan sebagai berikut :

Sumber Clock	Jumlah pulsa/sumber clock	Keterangan
clkI/O/1	12.000.000/1 = 12000.000	Melebihi batas nilai 16 bit
clkI/O/8	12.000.000/8 = 1.500.000	Melebihi batas nilai 16 bit
clkI/O/64	12.000.000/64 = 187.500	Melebihi batas nilai 16 bit
clkI/O/256	12.000.000/256 = 46.875	Masuk dalam nilai 16 bit
clkI/O/1024	12.000.000/1024 = 11.718,75	Masuk dalam nilai 16 bit

Dari table hasil pembagian di atas tampak bahwa sumber clock yang dapat dipakai adalah sumber clock dengan prescaler pembagi 256 dan 1024. Tetapi karena pre scaler 256 menghasilkan angka bulat tanpa nilai lebih dibelakang koma, maka sumber clock ini akan menghasilkan timer yang tepat pula. Oleh karena itu dipilihlah nilai prescaler ini dengan hasil pembagian 46.875

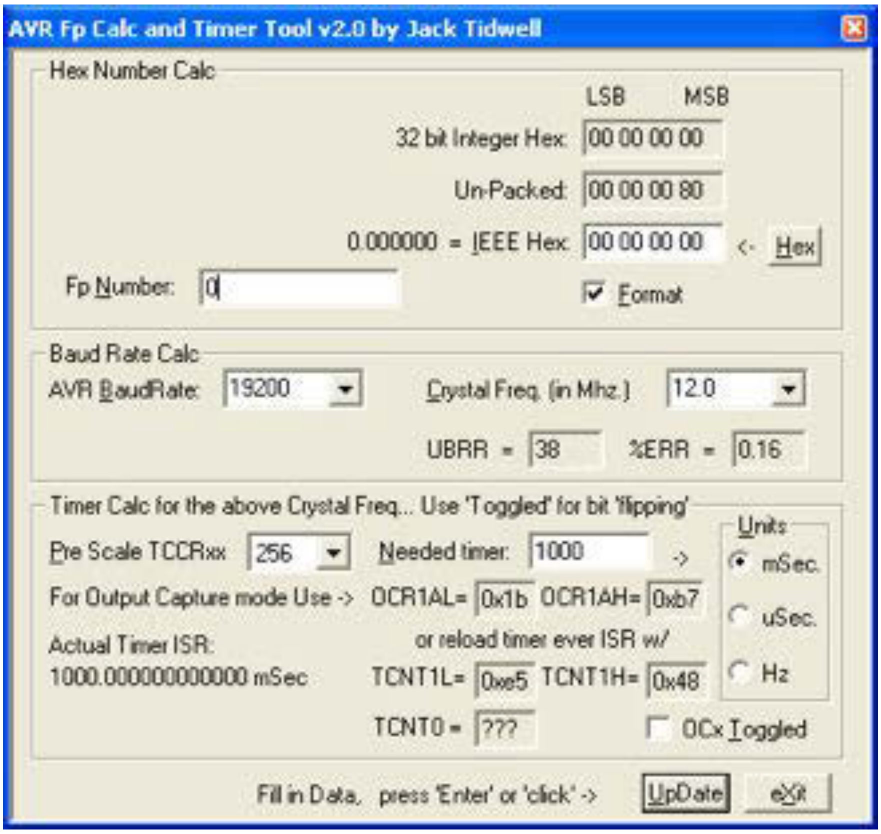


3. Berikutnya menghitung nilai yang akan dimasukkan ke register Timer/Counter1 – TCNT1H dan TCNT1L



Karena Timer menghitung naik dan over flow jika melewati nilai $FFFF ke $0000, maka register TCNT1 diisi dengan nilai minus - 46.875 atau dalam bilangan heksa $48E5. Nilai byte tinggi 48 inilah yang nantinya dimasukkan ke register TCNT1H dan Nilai byte rendah E5 dimasukkan ke register TCNT1L

Untuk mendapatkan nilai-nilai tersebut dalam algoritma langkah 2 dan 3, lebih mudah dilakukan dengan menggunakan kalkulator yang disertakan dalam CD dengan tampilan sebagai berikut :



Gambar 4.13 Kalkulator AVR

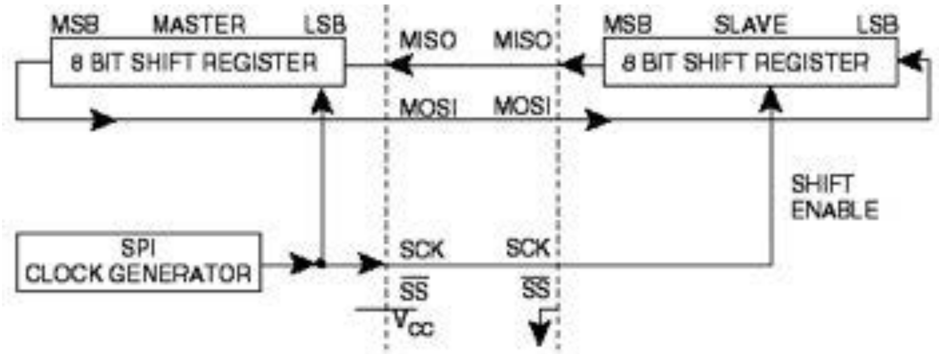


4. Flag overflow dapat dilihat pada bit TOV1 pada register TIFR. Apabila bit TOV berlogika 1, maka timer/counter sudah overflow dan program bagian (sub routine) tunda waktu selesai dan kembali ke program utama. Sebelum keluar sub routine bit TOV harus dienolkan dulu dengan memberikan logika satu.

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TOV1 otomatis di-clear ketika Timer/Counter1 Overflow Interrupt Vector diekesekusi. Alternatif lain TOV1 juga dapat di-clear dengan memberikan logika 1 pada lokasi bit tersebut.

4.7 Serial Peripheral Interface – SPI



Gambar 4.14 *SPI Master-Slave Interconnection*

Dengan adanya *Serial Peripheral Interface* – SPI memungkinkan transfer *sinkron* data kecepatan tinggi antara Atmega16 dengan peralatan lain. Hubungan interkoneksi antara *Master* dan *Slave* ditunjukkan pada gambar di atas. Sistim terdiri dari dua register geser dan *generator clock master*. Untuk menentukan perangkat mana sebagai *Master* atau *Slave*, ditentukan dengan men-set pin *Slave Select* (SS). Jika pin SS mendapat logika rendah, maka konfigurasi SPI adalah sebagai *slave*.

Master harus berinisiatif memulai komunikasi ke Slave. Master dan slave mempersiapakan data yang akan dikirim ke register geser. Master menggenerasi sinyal clock (SCK) untuk menggesesr data. Data selalu bergeser dari Master ke Slave pada pin Master Out – Slave In (MOSI) dan dari Slave ke Master pada pin



Master In – Slave Out (MISO). Setelah setiap paket terkirim, Master akan mensinkronisasi Slave dengan mem-pulling high Slave Select (SS).

Ketika dikonfigurasi sebagai Master, Interface SPI tidak secara otomatis jalur SS ini. Pemakai harus mengatur logika pin SS ini dengan dengan software sebelum komunikasi dapat dimulai. Ketika hal tersebut sudah dilakukan, tulis data ke SPI Data Register dan berikan pulsa clock, maka hardware akan menggeser delapan bit data ke Slave. Setelah menggeser satu byte, SPI clock generator akan stop, dan menyeting End of ransmission Flag (SPIF). Jika SPI *Interrupt Enable* bit (SPIE) dalam Register SPCR sama dengan 1, maka suatu *interrupt* akan di-*request*. Master dapat melanjutkan pengiriman data berikutnya dengan menulisi register SPDR, atau sinyal tanda akhir paket dengan mem-*pulling high pin Slave Select* (SS). Byte yang datang terakhir akan disimpan dalam register *buffer* untuk penggunaan selanjutnya.

Ketika dikonfigurasi sebagai Slave, Interface SPI akan beristirahat bersama MISO tri-state selama pin SS berlogika tinggi. Dalam kondisi ini, software dapat meng-update isi register SPI Data Register (SPDR), tetapi data tidak akan digeser keluar oleh pulsa clock yag datang pada pin SCK sampai pin SS diberi logika rendah. Setelah satu byte data selesai digeser, *End of Transmission Flag*, SPIF akan di-set 1. Jika bit SPI *Interrupt Enable*, SPIE dalam register SPCR di set 1, maka suatu interrupt akan di-request. Slave dapat melanjutkan menempatkan data baru yang akan dikirim ke register SPDR sebelum membaca data yang masuk. Byte data yang masuk terakhir akan disimpan ke dalam register buffer untuk penggunaan selanjutnya.

Sistim SPI ini menggunakan satu buah buffer untuk arah kirim dan dua buah buffer untuk arah terima. Artinya bahwa byte yang akan dikirim tidak dapat dimasukkan ke register data SPI sebelum siklus geser selesai. Ketika menerima data. Bagaimanapun juga sebuah karakter harus dibaca dari register data SPI sebelum karakter berikutnya selesai digeser, kalau tidak kehilangan byte pertama.

Dalam mode SPI Slave, control ligic akan mengambil sample sinyal dari pin SCK. Untuk meyakinkan sampling yang benar dari sinyal clock, minimal dan maksimal perioda harus sebagai berikut :



Periode rendah = lebih panjang dari 2 siklus clock CPU

Periode tinggi = lebih panjang dari 2 siklus clock CPU

Ketika SPI di-enable-kan, arah data pin MOSI, MISO, SCK dan SS disesuaikan dengan tabel berikut:

Pin	Arah, SPI Master	Arah , SPI Slave
MOSI	Ditentukan pemakai	input
MISO	input	Ditentukan pemakai
SCK	Ditentukan pemakai	input
$\overline{SS}$	Ditentukan pemakai	input

Contoh berikut memeprlihatkan bagaimana menginisialisasi SPI sebagai Master dan bagaimana mengirimkan data. DDR_SPI pada contoh ini harus diganti dengan Data Direction Register yang mengontrol pin SPI yang sesungguhnya. DD_MOSI, DD_MISO dan DD_SCK harus diganti dengan bit arah data yang sesungguhnya dari pin tersebut. Sebagai contoh, jika MOSI berada pada pin PB5, maka gantilah DD_MOSI dengan DDB5 dan DDR_SPI dengan DDRB.

```
SPI_SlaveInit:
; Set MISO output, all others input
ldi    r17,(1<<DD_MISO)
out    DDR_SPI,r17
; Enable SPI
ldi    r17,(1<<SPE)
out    SPCR,r17
ret

SPI_SlaveReceive:
; Wait for reception complete
sbis   SPSR,SPIF
rjmp   SPI_SlaveReceive
; Read received data and return
in     r16,SPDR
ret
```



Rangkuman

Mikrokontroller Atmega16 adalah mikrokontroller AVR 8 bit buatan ATMEL yang memiliki arsitektur RISC (Reduce Instruction Set Computing). Instruksi dikemas dalam kode 16 bit dan dijalankan hanya dengan satu siklus clock.

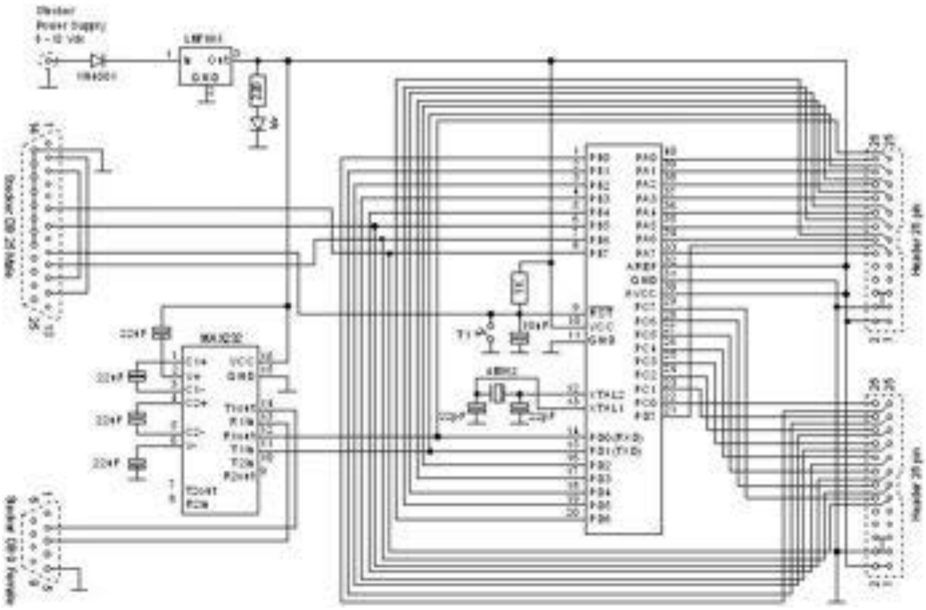
Fitur Mikrokontroller Atmega16 adalah

- High-performance, Low-power AVR® 8-bit Microcontroller
- Nonvolatile Program and Data Memories
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
  - Byte-oriented Two-wire Serial Interface
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad QFN/MLF
- Operating Voltages
  - 2.7 - 5.5V for Atmega16L
  - 4.5 - 5.5V for Atmega16
- Speed Grades
  - 0 - 8 MHz for Atmega16L
  - 0 - 16 MHz for Atmega16

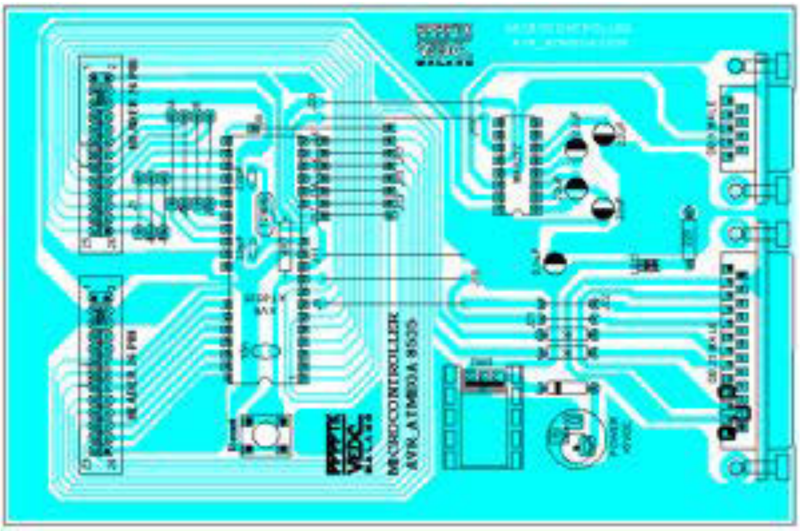


Latihan

Dengan menggunakan software Eagle atau Protel, gambarlah rangkaian dan layout PCB modul mikrokontroller Atmega16 seperti gambar berikut



Gambar 4.15 Rangkaian Modul Mikrokontroller Atmega16

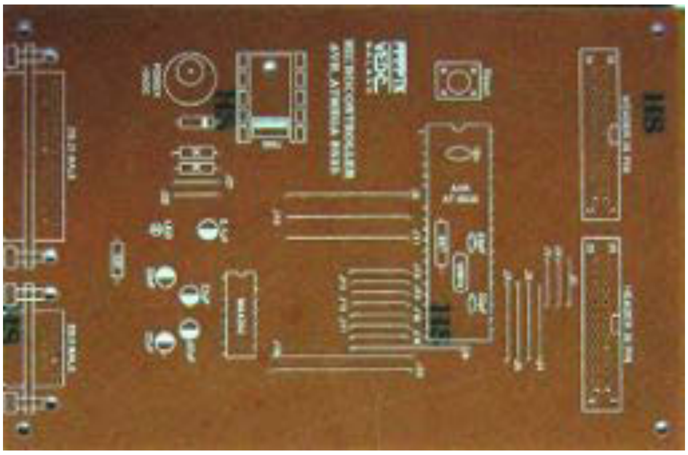


Gambar 4.16 Desaign PCB Modul Mikrokontroller Atmega16

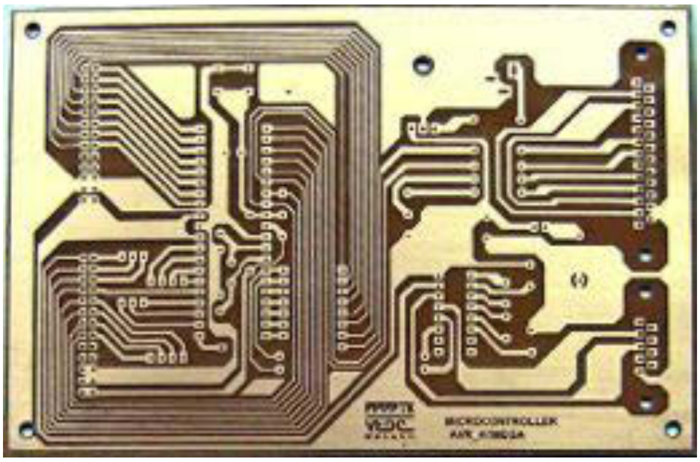
Tugas

Dari gambar layout PCB modul mikrokontroller yang dibuat pada latihan di atas, kerjakanlah pembuatan PCB tersebut dan solderilah komponen yang diperlukan sehingga menjadi modul mikrokontroller yang nantinya akan dipakai pada modul pembelajaran ini.

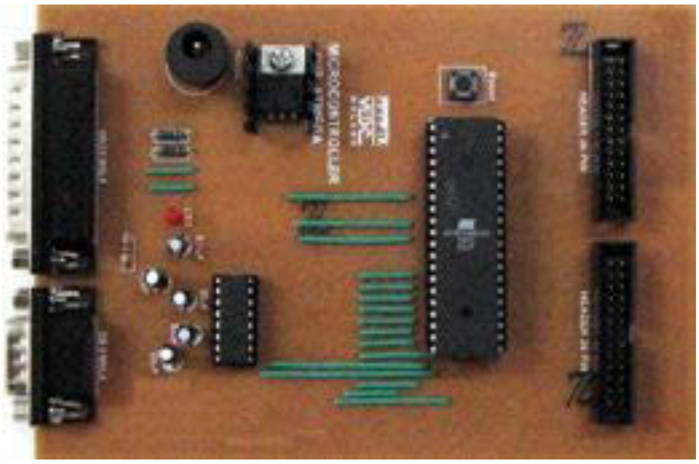
Kunci Jawaban



Gambar 4.17 Layout PCB sisi komponen



Gambar 4.18 Layout PCB sisi solder



Gambar 4.19 PCB Modul Mikrokontroller



KEGIATAN 2

Dasar BahasaBASIC untuk Pemrograman Mikrokontroller

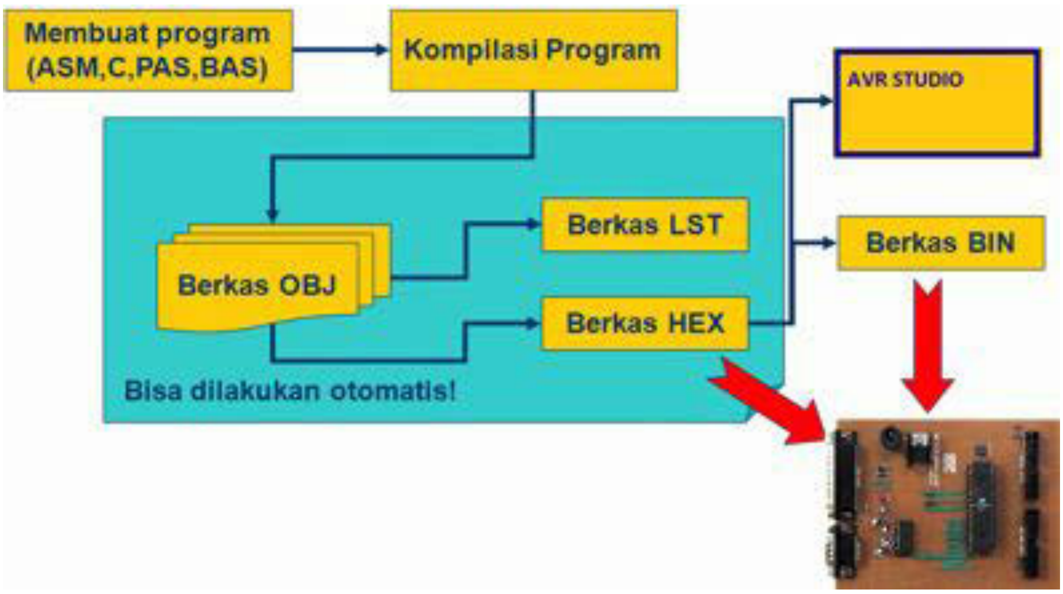
Tujuan Kegiatan Pembelajaran

Setelah mengikuti kegiatan pembelajaran pada pokok bahasan ini, diharapkan peserta didik dapat memiliki kemampuan membuat program, mensimulasikan dan mempogram mikrokontroller Atmega16 menggunakan software Bascom-AVR

4.8 Membuat Program Mikrokontroller

Pembuatan program aplikasi bisa dilakukan dengan bahasa pemrograman apapun yang memungkinkan. Untuk seorang siswa pemula, di sarankan untuk menggunakan **BASCOM 8051** atau **BASCOM AVR** dari **MCSELECTRONICS.COM**, karena sangat mudah penggunaanya.

Program bahasa apapun yang digunakan, harus dikompilasimenjadi berkas **BIN** atau**HEX** (format heksadesimal), sebagaimana prosesnya ditunjukkan pada Gambar dibawah ini. Kemudian di-*download*-kan ke mikrokontroler yang bersangkutan. Sehingga mikrokontroler Anda bisa menjalankan program tersebut.





Gambar 4.20. Alur pemrograman aplikasi mikrokontroler

Langkah selanjutnya tentu saja **MELAKUKAN UJI-COBA DAN EVALUASI** apakah rangkaian dan program sudah berjalan dengan benar atau belum, lakukan ***TROUBLESHOTING***jika memang masih ada kesalahan mayor maupun minor, sehingga hasil yang diperoleh menjadi baik dan benar.

Apakah ada masalah dengan program? Cek pada program Anda. Atau ada masalah pada rangkaian? Silahkan cek, apakah rangkaian sudah benar. Atau justru permasalahan terjadi karena Anda memberikan solusi yang salah atau kurang tepat, silahkan cek semuanya. Kesalahan bisa terjadi pada **RANGKAIAN** dan/atau pada **PROGRAM**, bahkan seringkali kesalahan-kesalahan sepele tetapi berdampak besar pada jalannya aplikasi.

4.9 Bahasa Pemrograman BASIC AVR (BASCOM AVR)

Sebagaimana telah di jelaskan sebelumnya, banyak cara dalam menuliskan program ke mikrokontroler, salah satunya bahasa BASIC. Penggunaan bahasa ini mempunyai kemudahan dalam memprogram dan adanya fasilitas simulator pada kompailer BASCOM AVR.

4.9.1 Tipe Data

Tipe data berkaitan dengan peubah atau variabel atau konstanta yang akan menunjukkan daya tampung/jangkauan dari variabel/konstanta tersebut. Tipe data dalam BASCOM ditunjukkan pada Tabel 1.4.

Tabel 2.1. Tipe Data dan Ukurannya

Tipe Data	Ukuran (Byte)	Jangkauan Data
Bit	1/8	0 atau 1
Byte	1	0 s/d 255
Integer	2	-32768 s/d 3.767
Word	2	0 s/d 65535
Long	4	-2147483648 s/d 2147483647
Single	4	$1,5 \times 10^{-45}$ s/d $3,4 \times 10^{38}$
Double	8	5×10^{-324} s/d $1,7 \times 10^{308}$
String	s/d 254	



4.9.2 Variabel

Variabel digunakan untuk menyimpan data sementara. Variabel diberi nama dan dideklarasikan terlebih dahulu sebelum digunakan. Aturan pemberian nama variabel sebagaiberikut:

- Harus dimulai dengan huruf (bukan angka).
- Tidak ada nama variabel yang sama dalam sebuah program.
- Maksimum 32 karakter
- Tanpa menggunakan spasi, pemisahan bisa dilakukan dengan garis bawah.
- Tidak menggunakan karakter-karakter khusus yang digunakan sebagai operator BASCOM

Variabel dapat dideklarasikan dengan beberapa cara :

1. **Dengan pernyataan DIM**

Deklarasi ini dibuat dengan perintah**DIM** (singkatan dori dimension) dengan aturansebagai berikut:

```
Dim <NamaVariabel> As <TipeData>
```

Contoh :

```
Dim angka As Integer
Dim bilangan As byte
```

Jika beberapa variabel dideklarasikan dalam satu baris, maka harus dipisah dengantanda koma.

Contoh:

```
Dim angka As Integer, bilangan As byte
```

2. **Dengan pernyataan DEFINT, DEFBIT, DEFBYTE, DEFWORD**

Deklarasi dengan pernyataan tersebut secara prinsip tidak berbeda dengan "DIM",perhatikan keterangan dari masing-masing pendeklarasian tersebut:

DEFINT = untuk tipe data integer,

DEFBIT = untuk tipe data bit,

DEFBYTE = untuk tipe data byte,

DEFWORD = untuk tipe data word,

DEFLNG= untuk tipe data long,



DEFSNG= untuk tipe data singel,

DEFDBL = untuk tipe data doubel.

Cara pendeklarasianya sebagai berikut:

```
DEFINT/DEFBIT/DEFBYTE/DEFWORD <variabel>
```

Contoh :

```
DEFINT angka
DEFBYTE bilangan
```

Untuk variabel dengan tipe data yang sama dapat dideklarasikan dengan dipisah titikkoma, misal :

```
DEFINT bil_1 ; bil 2 ; bil 3
```



4.9.3 Konstanta

Berbeda dengan variabel, sebuah konstanta akan bernilai tetap. Sebelum digunakan, konstanta dideklarasikan terlebih dulu dengan cara (ada dua cara):

```
Dim nama_konstanta As const nilai_konstanta
Const nama_konstanta = nilai_konstanta
```

Contoh :

```
Dim pembagi as const 23

Const pembagi = 23
```

4.9.4. Penulisan Bilangan

Pada BASCOM-AVR, bilangan dapat ditulis dalam 3 bentuk :

1. Desimal ditulis biasa, contoh : 16
2. Biner diawali dengan &B, contoh : &B10001111
3. Heksadesimal diawali dengan &H, contoh : &H8F

4.9.5. Alias

Untuk mempermudah pemrograman, biasanya nama register dalam mikrokontroler dibuatkan nama yang identik dengan hardware yang dibuat, contoh :

```
LED_1 alias PORTC.0    ' nama lain dari PORTC.0 adalah LED_1

SW_1 alias PINC.1      ' nama lain dari PINC.1 adalah SW_1
```

4.9.6. Array atau Larik

Array atau larik merupakan sekumpulan variabel dengan nama dan tipe yang sama, yang berbeda indeks keanggotaannya.

Cara mendeklarasikan array sebagai berikut:

```
Dim nama array(jumlah anggota) as tipe_data
```



Contoh:

```
Dim A(8) as byte        ' variabel A dengan tipe data byte
                        ' dengan 8 anggota
```

Untuk mengakses array dengan cara :

```
A(1) = 25               ' anggota pertama variabel A isinya 25
PORTC=A(1)              'PORTC = nilai variabel A(1) = 25
```

4.9.7. Operator Matematika dan Logika

Operator digunakan dalam pengolahan data pemrograman dan biasanya membutuhkan dua variabel atau dua parameter, sedangkan operator dituliskan di antara kedua parameter tersebut. Operator-operator BASCOM AVR ditunjjukkan pada Tabel 2.2, Tabel 2.3 dan Tabel 2.4.

Tabel 2.2. Operator Aritmetik.

Operator	Keterangan	Contoh
+	Operasi penjumlahan	A + B
-	Operasi pengurangan	A - B
*	Operasi perkalian	A * B
/	Operasi pembagian	A / B
%	Operasi sisa pembagian	A % B

Tabel 2.3. Operator Relasional

Operator	Keterangan	Contoh
=	Sama dengan	A = B
<>	Tidak sama dengan	A <> B
>	Lebih besar dari	A > B
<	Lebih kecil dari	A < B
>=	Lebih besar atau sama dengan	A >= B
<=	Lebih kecil atau sama dengan	A <= B

Tabel 2.4. Operator Logika

Operator	Keterangan	Contoh
AND	Operasi AND	&B110 And &B101 = &B100
OR	Operasi OR	&B11001 Or &B10111 = &B11111
NOT	Operasi NOT	NOT &HFF = &HO
XOR	Operasi XOR	&B1001 Xor &B0111 = &B1110



4.9.8. Operasi Bersyarat

A. IF – THEN

Sebuah atau serangkaian instruksi akan dikerjakan jika memenuhi syarat-syarat atau kondisi tertentu. Cara penulisannya sebagai berikut :

```
If <kondisi> Then <perintah>       ` 1 baris perintah
If <kondisi> Then                  ` lebih dari 1 perintah
<perintah 1>
<perintah 2>
...
End If
```

B. IF - THEN – ELSE

Versi lengkap dari sebuah atau serangkaian instruksi akan dikerjakan jika memenuhi syarat-syarat atau kondisi tertentu, jika tidak dipenuhi maka instruksi atau serangkaian instruksi lainnya-lah yang akan dikerjakan. Cara penulisannya sebagai berikut:

```
If <kondisi> Then
<perintah 1>
...
Else
<perintah 2>
...
End If
```

C. IF - THEN - ELSEIF

Sama seperti IF-THEN-ELSE, hanya jika kondisi tidak dipenuhi masih dilakukan pengujianapakah suatu kondisi memenuhi syarat lainnya. Cara atau sintaks (*syntax*) penulisannyasebagai berikut:

```
If <kondisi 1> Then
<perintah 1>
...
Elseif <kondisi 2> Then

<perintah 2>
...
End If
```



D. SELECT - CASE

Cocok digunakan untuk menangani pengujian kondisi yang jumlahnya cukup banyak. Cara penulisannya :

```
Select case <variabel>
      Case 1: <perintah 1>
      Case 2: <perintah 2>
      ...
End Select
```

4.10. Operasi Pengulangan

A. FOR - NEXT

Perintah ini digunakan untuk melaksanakan perintah secara berulang sesuai dengan jumlah yang ditentukan. Sintaks penulisannya :

```
For <var> = <nil_awal> To <nil_akhir><step angka>
      <perintah>
Next [<var>]
```

B. DO - LOOP

Pernyataan ini untuk melakukan perulangan terus menerus tanpa henti (pengulangan tak berhingga) selama mikrokontroler-nya masih mendapatkan detak dan/atau catu daya. Cara penulisannya :

```
Do
      <pernyataan>
      ...
Loop
```

Jika pengulangan dibatasi oleh suatu kondisi maka caranya ditunjukkan berikut ini, artinya pengulangan terus dilakukan sehingga suatu kondisi terpenuhi atau melakukan pengulangan selama kondisinya salah:

```
Do
      <pernyataan>
      ...
Loop Until <kondisi>
```



C. WHILE - WEND

Berbeda dengan DO-LOOP, instruksi ini digunakan untuk melakukan pengulangan selama kondisinya benar, cara penulisannya:

```
While <kondisi>
      <perintah>
      ...
Wend
```

4.11. Lompatan Proses

A. GOSUB <nama_subrutin>

Perintah ini akan melakukan lompatan sebuah subrutin, kemudian kembali lagi setelahsubrutin perintah tersebut selesai dikerjakan. Rutin yang dibuat harus diakhiri denganinstruksi **RETURN**. Contoh:

```
Print "We will start execution here"
Gosub Routine
Print "Back from Routine"
End

Routine:
Print "This will be executed"
Return
```

B. GOTO <label>

Perintah ini untuk melakukan lompatan ke label kemudian melakukan serangkaian instruksi tanpa harus kembali lagi, sehingga tidak perlu **RETURN**. Contoh:

```
Dim A As Byte

Start:                    'sebuah label diakhiri dengan :
A = A + 1                 'naikkan variabel A
If A < 10 Then            'apakah lebih kecil 10?
      Goto Start          'ya, lakukan lagi
End If                    'akhir IF

Print "Ready"             'ok
```



C. EXIT

Untuk keluar secara langsung dari perulangan DO-LOOP, FOR-NEXT, WHILE-WEND. Carapenulisannya sebagai berikut :

```
EXIT FOR          (keluar dari For-Next)
EXIT DO           (keluar dari Do-Loop)
EXIT WHILE        (keluar dari While-Wend)
EXIT SUB          (keluar dari Sub-Endsub)
EXIT FUNCTION     (keluar dari suatu fungsi)
```



Rangkuman

BASCOM AVR merupakan software yang digunakan untuk memprogram mikrokontroller keluaran dari MCS ELECTRONIC.COM. Penggunaan BASCOM AVR relatif sangat mudah dipahami oleh siswa pemulakarena menggunakan bahasa pemrograman tingkat tinggi BASIC.

Dalam mempelajari bahasa pemrograman mikrokontroller, maka dasar utama yang harus dipegang oleh setiap siswa adalah beberapa poin sebagai berikut:

- Tipe data
- Variabel dan konstanta
- Deklarasi Variabel dan konstanta
- Operasi matematika dan logika
- Penulisan bilangan
- Konfigurasi input-output
- Operasi bersyarat (IF-THEN, ELSE IF, SWITCH CASE)
- Operasi perulangan (DO-LOOP, FOR-NEXT, WHILE-WEND)



Tugas

Perhatikan dan pelajari setiap instruksi syntax program dan perhatikan setiap aplikasi penggunaan syntaks tersebut dalam program!

Tes Formatif

1. Sebutkan tipe-tipe data yang digunakan dalam pemrograman mikrokontroller dengan menggunakan BASCOM!
2. Jelaskan dan berikan contoh saat kapan kita menggunakan variabel dengan bertipe single?
3. Sebutkan perbedaan dan keuntungan menggunakan SWITCH-CASE dibanding IF-THEN!
4. Jelaskan secara singkat perbedaan DO-LOOP dengan WHILE-WEND



KEGIATAN 3

Menntransfer Program Kedalam Mikrokontroller

Tujuan Kegiatan Belajar

Setelah mengikuti kegiatan pembelajaran pada pokok bahasan ini, diharapkan peserta didik dapat memiliki kemampuan menggunakan software BASCOM untuk memprogram, mensimulasikan dan mendownloadkan hasil kompile dari sfotware tersebut kedalam mikrokontroller ATMega16.

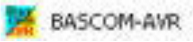
4.12 Membuat Program Mikrokontroller

Siapkan modul mikrokontroller dan sambungkan kabel LPT antara mikrokontroller dengan komputer.



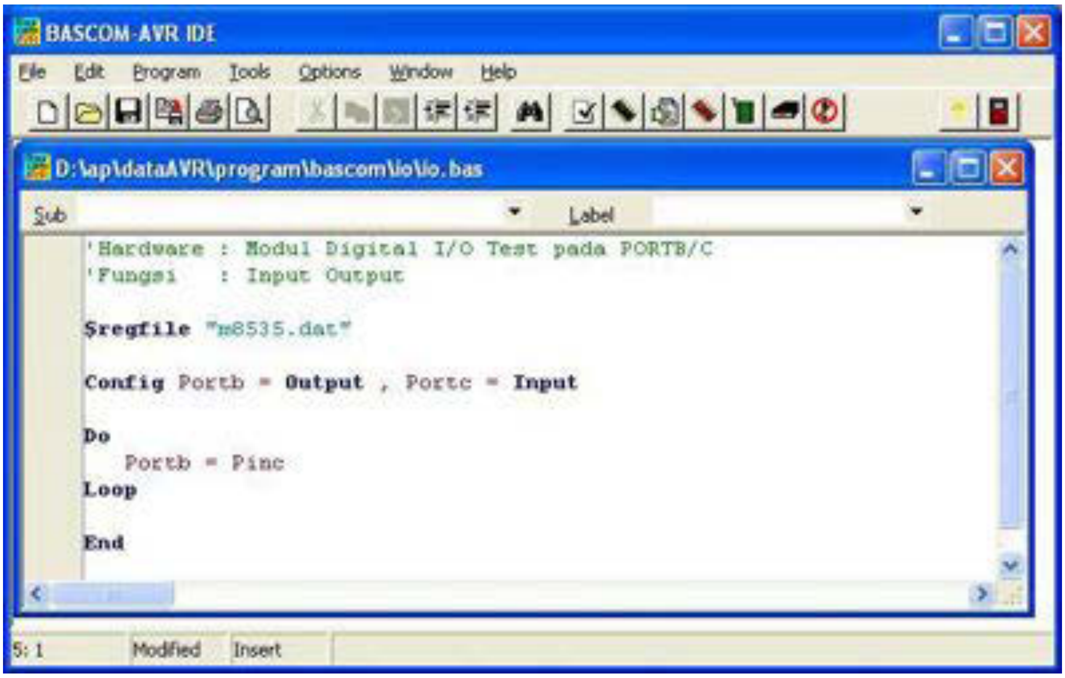
Gambar 3.1 Sambungan kabel LPT modul mikrokontroller dengan komputer



Jalankan program  , tunggu sampai muncul jendela utama BASCOM-AVR. Pilih Menu **File** – **New** atau tekan toolbar **Open new edit window**



Pada jendela editor, ketiklah program mikrokontroller yang akan dibuat. Di bawah ini contoh program input output membaca deretan saklar pada Pinc dan menampilkannya pada Portb.



Gambar 3.2 Jendela Editor BASCOM-AVR

Simpan file tersebut dalam satu folder tersendiri karena setiap project setelah di-compile akan menghasilkan banyak file.

Untuk menyimpan file, pilih menu **File** – **Save** atau tekan toolbar **Save File**





Kemudian compile file tersebut dengan memilih menu **Program** – **File** atau tekan toolbar **Compile current file(F7)**

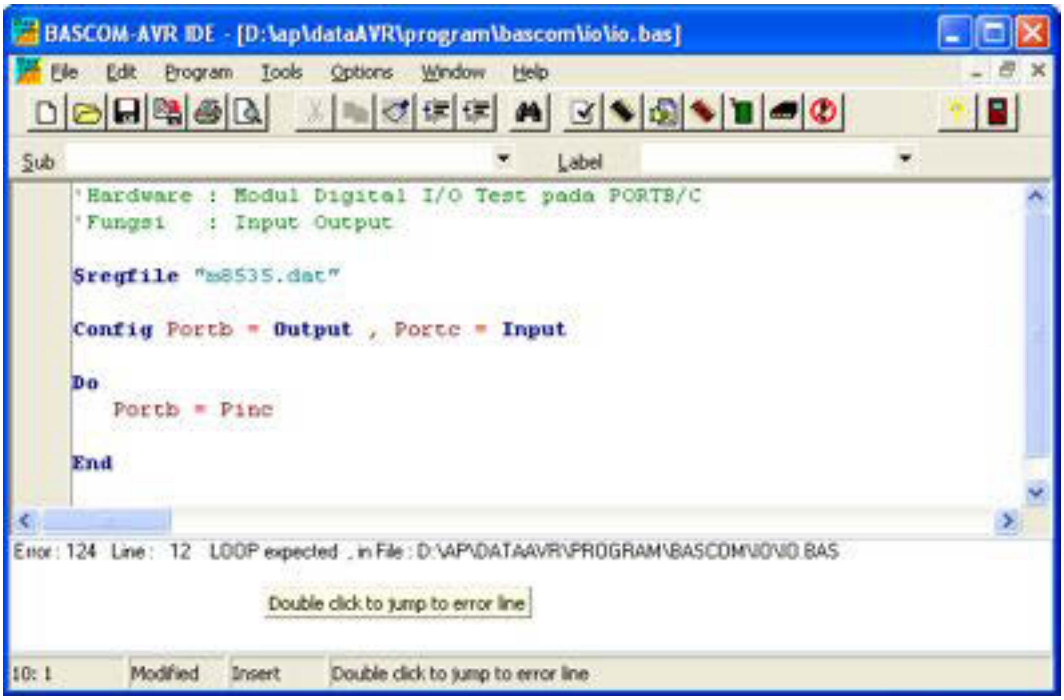


Tunggu sampai proses compiling selesai.



Gambar 3.3 Jendela Proses Compiling

Apabila terdapat kesalahan, maka di bawah jendela editor akan muncul informasi kesalahan seperti contoh berikut



Gambar 3.4 Jendela informasi kesalahan



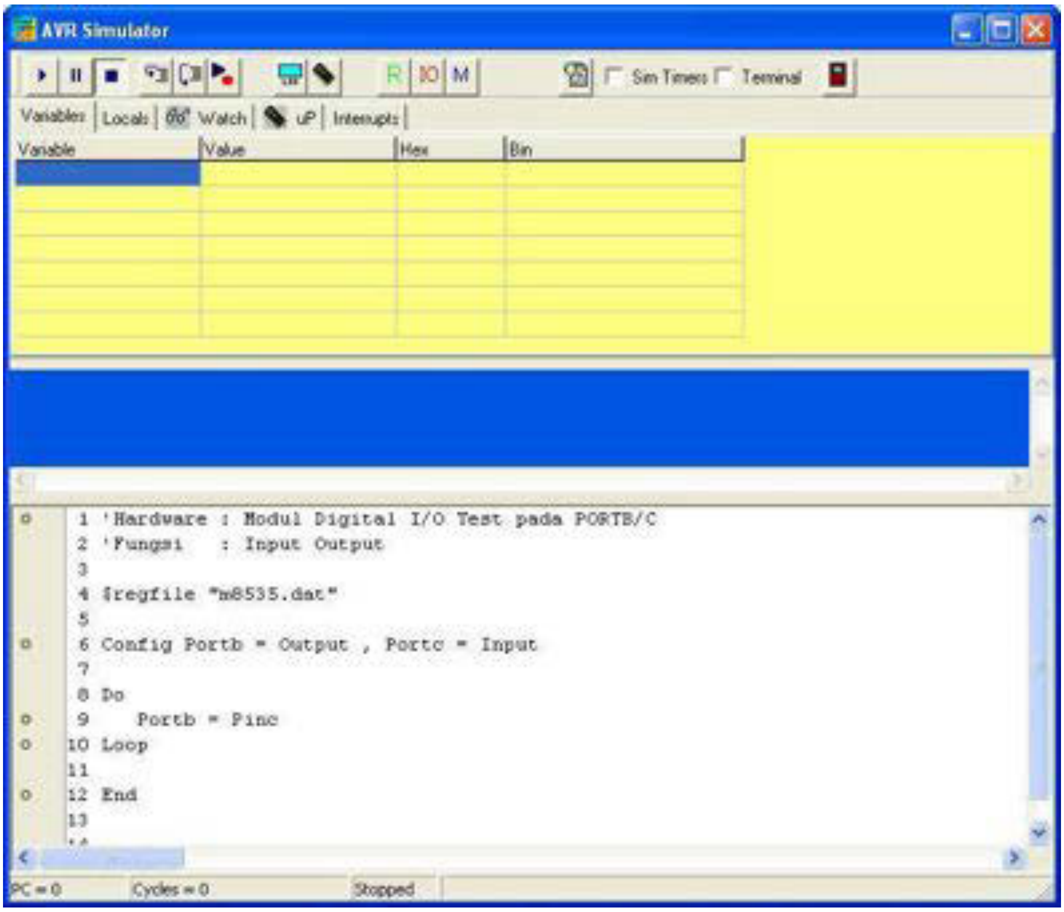
Tampak pada contoh di atas muncul pesan keslahan Error 124 pada baris 12 bahwa tidak ada instruksi LOOP. Untuk memperbaiki kesalahan yang dimaksud, lompat ke baris yang salah dengan cara double click pada teks informasi kesalahan tersebut.

4.13 Mensimulasikan Program Mikrokontroller

Jika tidak ada kesalahan, maka pesan kesalahan tidak muncul dan program dapat disimulasikan dengan cara pilih menu **Program** – **Simulate** atu tekan toolbar **Simulate program(F2)**



Selanjutnya akan muncul jendela AVR Simulasi seperti berikut

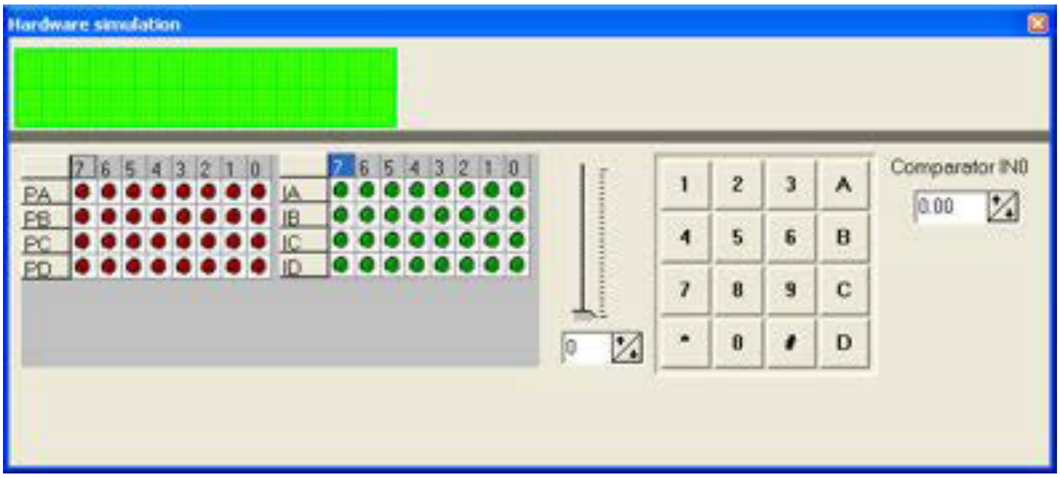


Gambar 3.5 Jendela AVR Simulasi



Aktifkan toolbar **Show hardware emulation** kemudian akan muncul Jendela Hardware Simation seperti gambar 3.6





Gambar 3.6 Jendela Hardware Simulasi

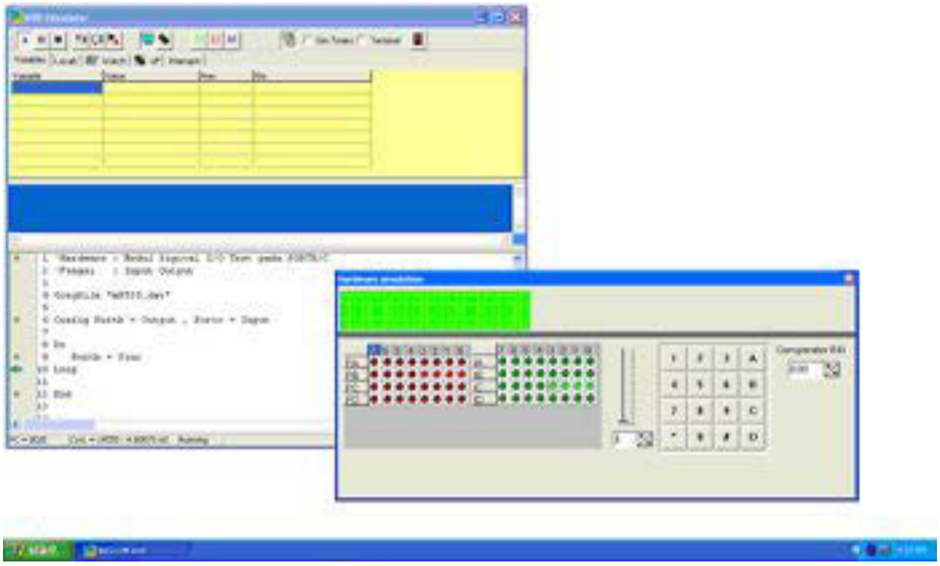
Aktifkan toolbar **Refresh variables** agar tampilan jendela hardware simulasi selalu fresh sesuai kondisi aktual



Berikutnya jalankan program simulasi dengan menekan toolbar **Run program(F5)**



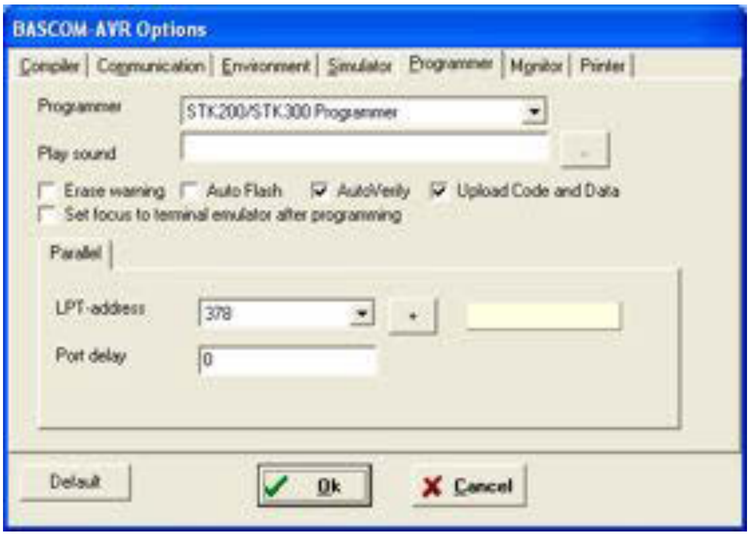
Klik pada tampilan LED warna hijau IC maka tampilan LED warna merah pada PB akan menyala sesuai masukan pada PinC.



Gambar 3.7 Jendela AVR Simulasi dan Hardware Simulasi keduanya aktif

4.14 Memprogram Mikrokontroller

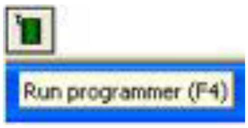
Pilih jenis programmer melalui port paralel LPT dengan cara pilih menu **Option** – **Programmer**, selanjutnya pilih TabStrib **Programmer** dan pada ComboBox **Programmer** pilih **STK200/STK300 Programmer**, kemudian tekan tombol **OK** seperti pada gambar berikut



Gambar 3.8 Jendela BASCOM-AVR Option



Selanjutnya pilih menu **Program – Send to chip** atau tekan toolbar **Run programmer (F4)** dan pilih menu **Program**

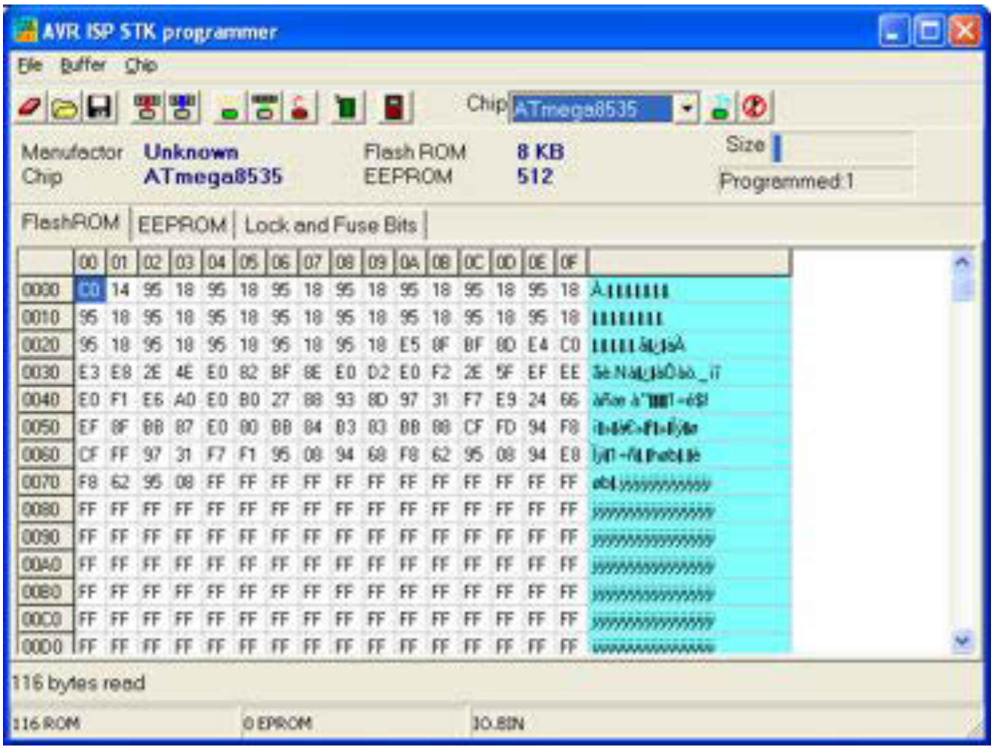


Apabila board modul mikrokontroller tidak aktif, maka akan muncul jendela pesan sebagai berikut



Gambar 3.9 Kotak pesan

Jika board modul mikrokontroller tidak ada masalah, maka selanjutnya akan muncul jendela AVR ISP STK Programmer sebagai berikut



Gambar 3.10 Jendela AVR ISP STK Programmer

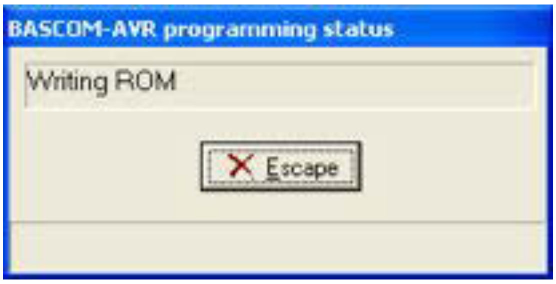
Pada combobox Chip, pilih **Atmega16**, tekan toolbar Load file into buffer dan pilih file hex yang akan di-download ke chip mikrokontroller



Untuk men-download program , tekan toolbar **Write buffer to flash ROM**



Tunggu sampai proses programming selesai



Gambar 3.11 Jendela BASCOM-AVR Programming status

Setelah itu lepas kabel LPT dan mikrokontroller langsung menjalankan program yang telah di-download



Rangkuman

Software BASCOM-AVR dapat dipergunakan untuk membuat program dengan bahasa tingkat tinggi BASIC.

Program yang sudah ditulis dapat di-compile dapat disimulasikan pada computer.

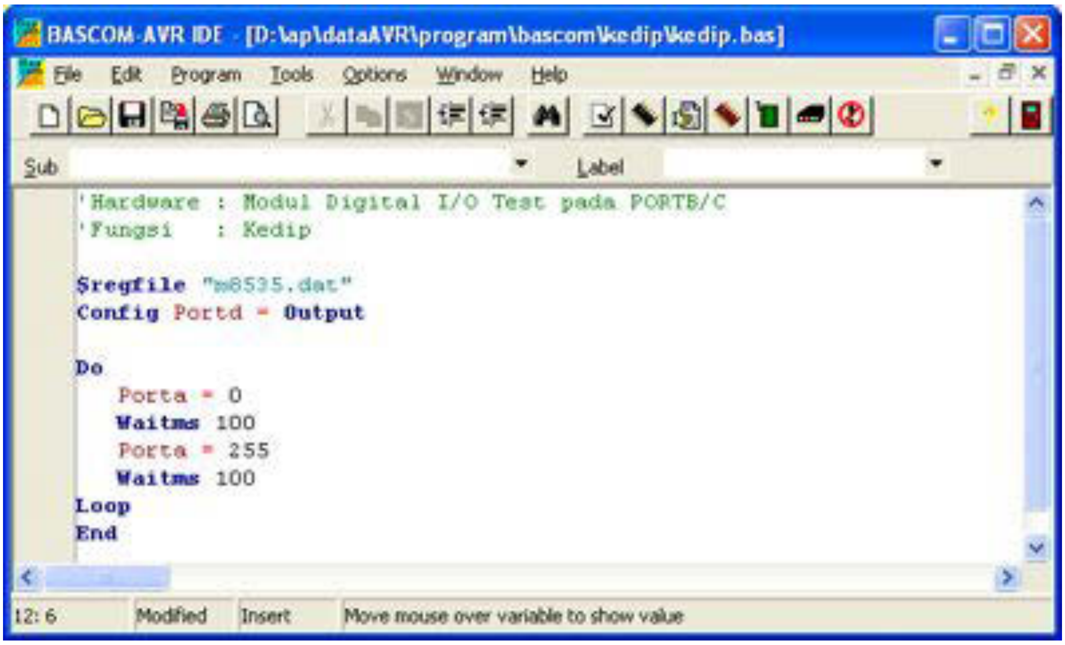
Hasil compiling program berupa file hex yang nantinya didownloadkan ke chip mikrokontroller.



Latihan

Salinlah program dibawah ini pada editor BASCOM-AVR, kemudian simulasikan dan downloadkanlah ke dalam chip microcontroller.

```
'Hardware : Modul Digital I/O Test pada PORTB/C
'Fungsi : Kedip
$regfile "m8535.dat"
Config Portd = Output
Do
   Portd = 0
   Waitms 100
   Portd = 255
   Waitms 100
Loop
End
```



Gambar 3.12 Program kedip pada jendela editor BASCOM-AVR



Tugas

Berikut ini adalah program membaca data analog ADC input kanal 0 dan ditampilkan ke LCD dua baris 16 kolom.

Salinlah program tersebut dan simulasikan pada BASCOM-AVR.

Aturlah slider mulai pada posisi minimal paling bawah sampai pada posisi paling atas. Berapa penunjukan LCD ketika slider pada posisi minimal dan maksimal ?

```
$regfile "m8535.dat"
$crystal = 4000000

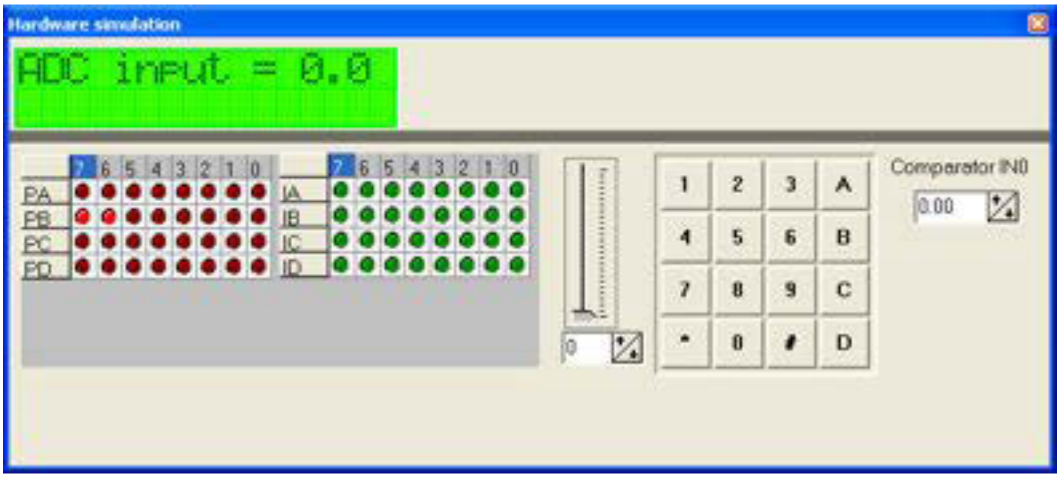
Dim Ch0 As Word
Dim A0 As Single

Config Adc = Single , Prescaler = Auto , Reference = Avcc
Config Lcdpin = Pin , Db4 = Portb.4 , Db5 = Portb.5 , Db6 =
Portb.6 , Db7 = Portb.7 , E = Portb.3 , Rs = Portb.2
Config Lcd = 16 * 2

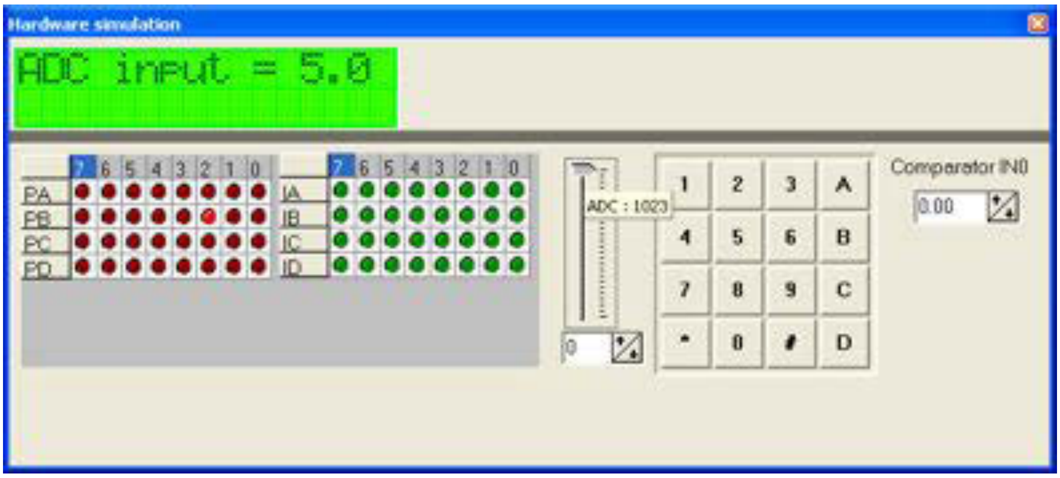
Cursor Off Noblink
Start Adc
Cls
Upperline
Lcd "ADC input = "

Do
  Ch0 = Getadc(0)
  A0 = Ch0 * 0.0049
  Locate 1 , 13
  Lcd Fusing(a0 , "#.#")
Loop
End
```

Kunci Jawaban



Gambar 3.13 Simulasi ketika slider pada posisi minimal



Gambar 3.14 Simulasi ketika slider pada posisi maksimal

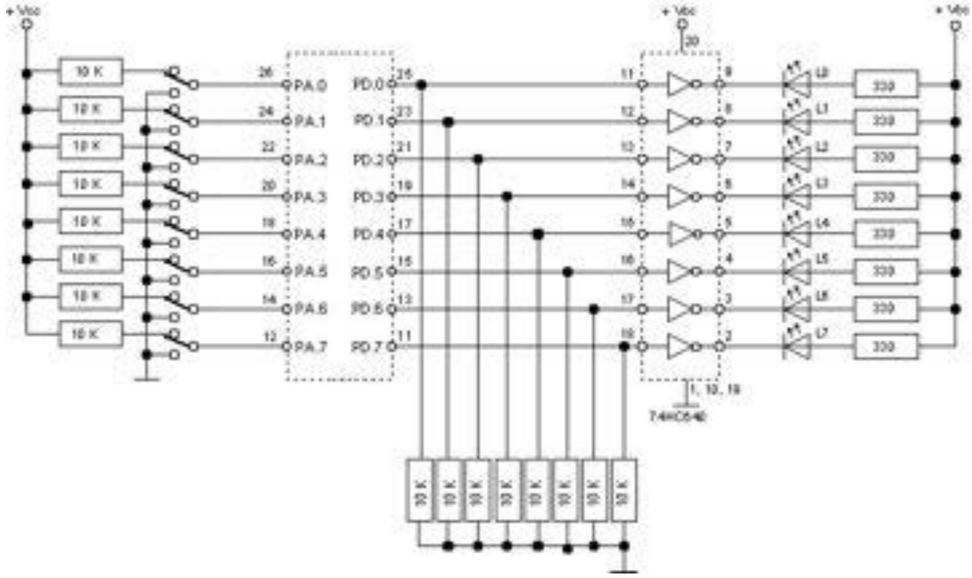
KEGIATAN 4

Aplikasi Pemrograman Mikrokontroller Menggunakan BASCOM

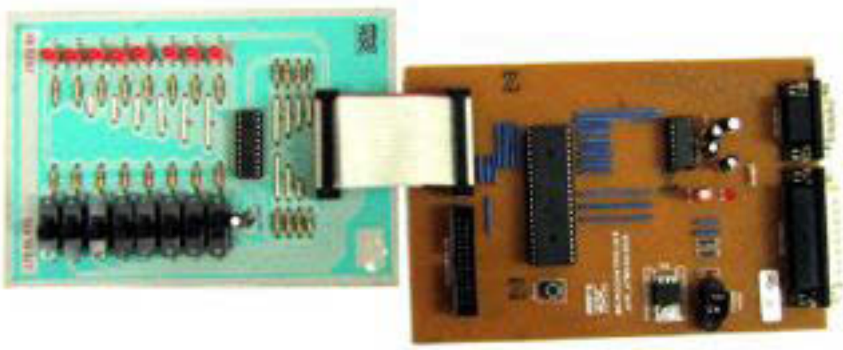
Tujuan Kegiatan Pembelajaran

Setelah mengikuti kegiatan pembelajaran pada pokok bahasan ini, diharapkan peserta didik dapat Memprogram Sistem Mikrokontroller ATMega16 kedalam aplikasi-aplikasi sederhana yang berhubungan dengan input-output.

4.15 Membuat Program Mikrokontroller



Gambar 4.1 Rangkaian Modul Digital Input Output Test





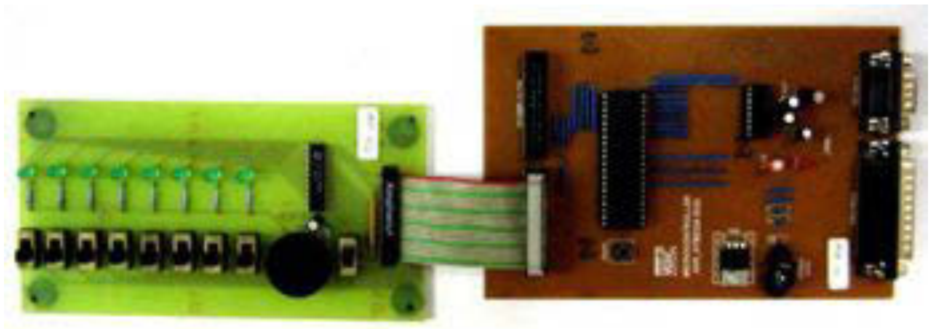
Gambar 4.2 Modul Percobaan Input Output Digital

Listing program :

```
'Hardware : Modul Digital I/O pada PORTD/PORTA
'Fungsi : Membaca data deretan saklar pada PORTA
' Menampilkan hasil pembacaan deretan LED pada PORTD
$regfile "m16def.dat "
Config Porta = Input
Config Portd = Output
Do
Portd = Pina
Loop
End
```



4.16 Deretan LED



Gambar 4.3 Modul Percobaan Deretan LED

Listing program :

Cara 1 :

```
'Hardware : Modul Mikrokontroler
' Modul Digital I/O pada PORTB/PORTC
'Fungsi : LED berjalan pada PORTB

$regfile " m16def.dat "
$crystal = 4000000

Config Portc = Input
Config Portb = Output

Do
   Portb = &B00000001
   Waitms 100
   Portb = &B00000010
   Waitms 100
   Portb = &B00000100
   Waitms 100
   Portb = &B00001000
   Waitms 100
   Portb = &B00010000
   Waitms 100
   Portb = &B00100000
   Waitms 100
   Portb = &B01000000
   Waitms 100
   Portb = &B10000000
   Waitms 100
Loop
End
```



Cara 2 :

```
‘Hardware : Modul Mikrokontroler
‘ Modul Digital I/O pada PORTB/PORTC
‘Fungsi : LED berjalan pada PORTB

$regfile " m16def.dat "
$crystal = 4000000

Dim Dat As Byte
Dim I As Single

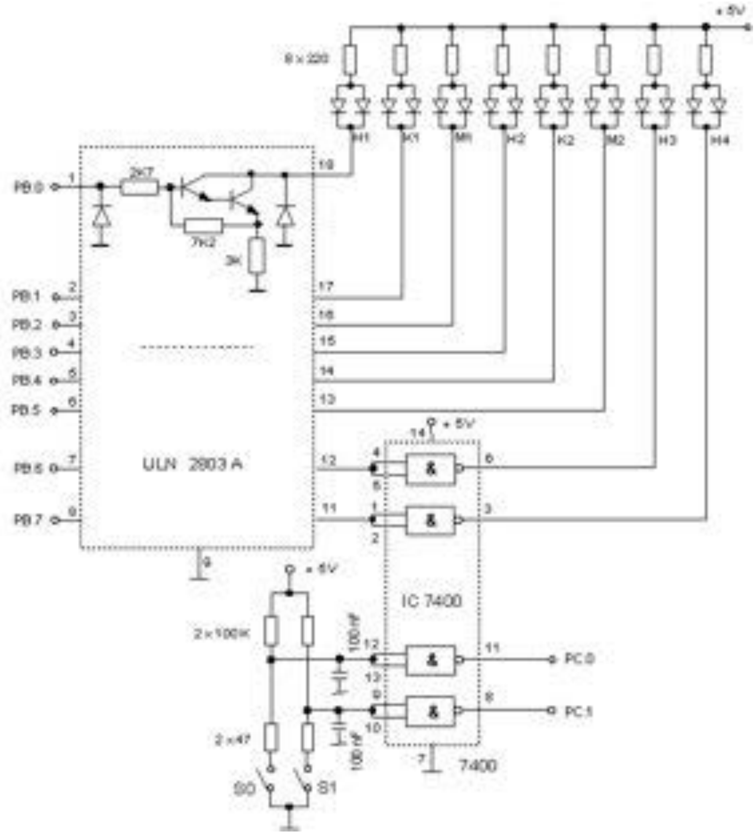
Config Portc = Input
Config Portb = Output

Do
Restore Teks
   For I = 1 To 8
     Read Dat
     Portb = Dat
     Waitms 100
   Next
Loop
End

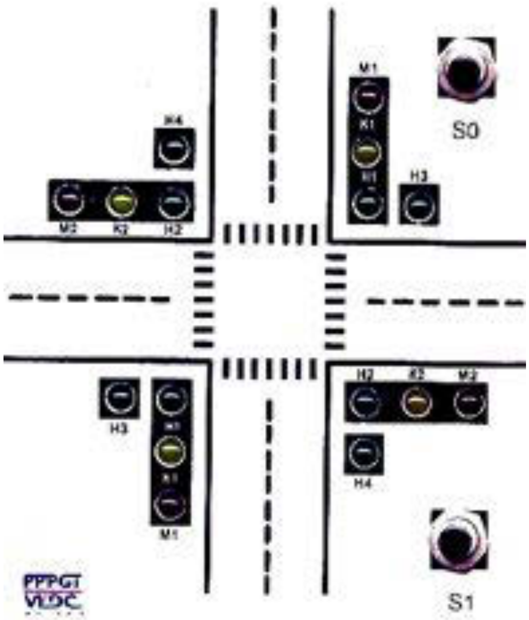
Teks:
Data &B00000001
Data &B00000010
Data &B00000100
Data &B00001000
Data &B00010000
Data &B00100000
Data &B01000000
Data &B10000000
```



4.17 Lampu Lalu Lintas

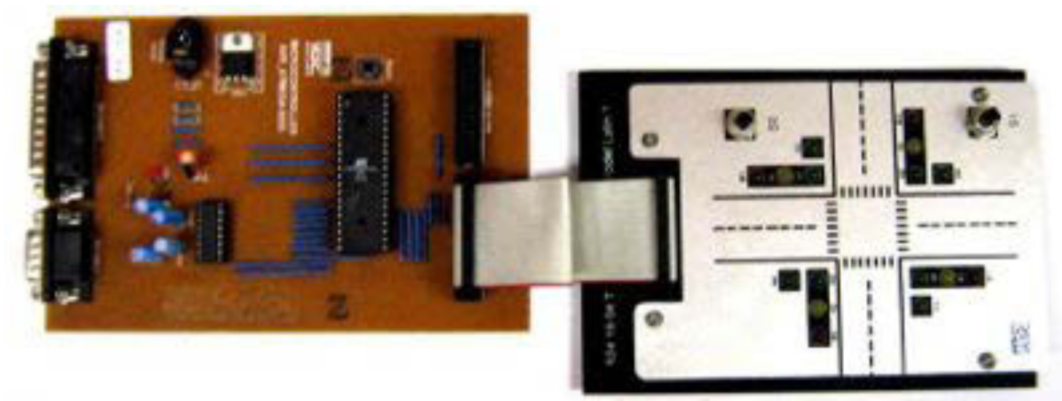


Gambar 4.4 Rangkaian Modul Lampu Lalu Lintas



Gambar 4.5 Penempatan LED pada Modul Lampu Lalu Lintas





Gambar 4.6 Modul Percobaan Lampu Lalu Lintas

Tabel kebenaran :

Lampu								Heksa	Penyalaan
H4	H3	M2	K2	H2	M1	K1	H1		
PB.7	PB.6	PB.5	PB.4	PB.3	PB.2	PB.1	PB.0		
1	1	1	0	0	0	0	1	&H21	5 detik
1	1	0	1	0	0	0	1	&H11	3 detik
1	1	0	0	1	1	0	0	&H0C	5 detik
1	1	0	0	1	0	1	0	&H0A	3 detik

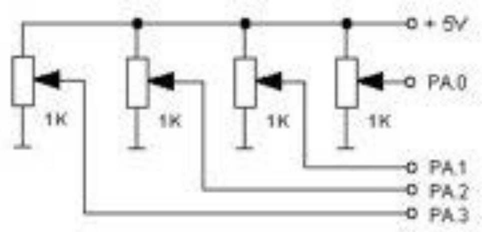
Listing program :

```
'Hardware : Modul Lampu Lalu Lintas pada PORTB/PORTC
'Fungsi : Pengaturan lampu lali lintas sesuai tabel
kebenaran

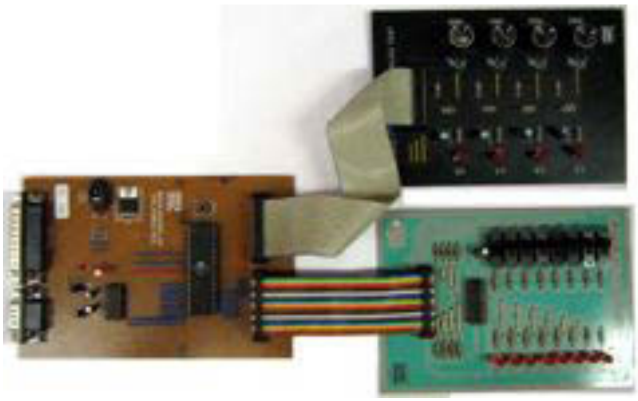
$regfile " m16def.dat "
$crystal = 4000000
Config Portc = Input
Config Portb = Output
Do
  Portb = &H21
  Wait 5
  Portb = &H11
  Wait 3
  Portb = &H0C
  Wait 5
  Portb = &H0A
  Wait 3
Loop
End
```



4.17 Analog To Digital Convertion (ADC)



Gambar 4.7 Rangkaian Modul Analog Input Test



Gambar 4.8 Modul Percobaan Analog Input Test

Listing program :

```
'Hardware : Modul Analog Input Test pada PORTA/PORTB
' Modul Digital Input Output Test pada PORTB/PORTC
'Fungsi : Membaca masukan analog PA.0, mengubah data 10 bit
'menjadi 8 bit dan menampilkannya pada PORTB

$regfile "m16def.dat"
$crystal = 4000000

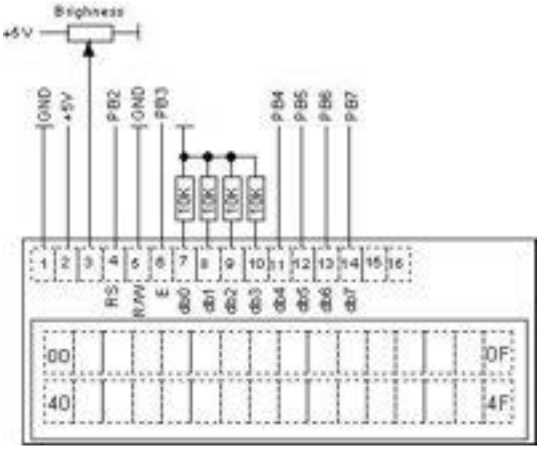
Dim A As Byte
Dim W As Word

Config Portb = Output
Config Portc = Input
Config Adc = Single , Prescaler = Auto
Start Adc

Do
  W = Getadc(0)
  W = W / 4
  A = W
  Portb = W
Loop
End
```



4.18 Liquid Crystal Display (LCD)



Gambar 4.9 Rangkaian Modul LCD



Gambar 4.10 Modul Percobaan LCD



Listing program :

```
'Hardware : Modul LCD pada PORTB + Modul Input Analog pada
PORTA
'Fungsi : Membaca data ADC 4 kanal
' Mengkonversi hasil pembacaan menjadi tampilan Volt,
' dengan rumus  A0  = Ch0 * 0.0049
'               Ch0 = 0 s.d.1023
'               A0  = 0 * 0.0049 = 0.0 Volt
'                   = 1023 * 0.0049 = 5.0127 -> 5.0 Volt
'tampilan format LCD  A0: x.x A2: x.x
'                     A1: x.x A3: x.x
$regfile "m16def.dat"
$crystal = 4000000

Declare Sub Baca_adc()
Declare Sub Tampil_lcd()

Dim Ch0 As Word , Ch1 As Word , Ch2 As Word , Ch3 As Word
Dim A0 As Single , A1 As Single , A2 As Single , A3 As
Single

Config Lcd = 16 * 2
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Config Lcdpin = Pin , Db4 = Portb.4 , Db5 = Portb.5 , Db6 =
Portb.6 , Db7 = Portb.7 , E = Portb.3 , Rs = Portb.2
Cursor Off Noblink

Start Adc
Cls
Upperline
Lcd "A0: A2: "
Lowerline
Lcd "A1: A3: "
Waitms 100

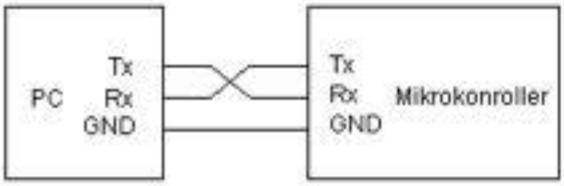
Do
  Call Baca_adc
  Call Tampil_lcd
  Waitms 100
Loop

Sub Baca_adc()
Ch0 = Getadc(0)
Ch1 = Getadc(1)
Ch2 = Getadc(2)
Ch3 = Getadc(3)
A0 = Ch0 * 0.0049
A1 = Ch1 * 0.0049
A2 = Ch2 * 0.0049
A3 = Ch3 * 0.0049
End Sub
```



```
Sub Tampil_lcd()
Locate 1 , 5
Lcd Fusing(a0 , "#.#")
Locate 2 , 5
Lcd Fusing(a1 , "#.#")
Locate 1 , 13
Lcd Fusing(a2 , "#.#")
Locate 2 , 13
Lcd Fusing(a3 , "#.#")
End Sub
End
```

4.19 Komunikasi Data Serial antara PC dengan Mikrokontroller Menggunakan USART



Gambar 4.11 Komunikasi serial antara PC dengan mikrokontroller



Gambar 4.12 Koneksi antara PC dengan mikrokontroller



Listing program :

```
'Hardware : Modul Digital Input Output Test pada PORTB/C
' Sambungkan kabel RS232 crossing ke PC
'Fungsi : Mengirim teks "Masukkan data PORTB : " ke PC
' Menunggu data masukan dari PC berupa angka 0 s.d 255
' Data yang diterima ditampilkan ke deretan LED pada
' PORTB
' Membaca data deretan saklar pada PORTC
' Mengirimkan teks ke PC "Data PORTC = "
' Menirimkan data deretan saklar PORTC ke PC berupa
' angka 0 s.d. 255

$regfile "m16def.dat"
$crystal = 4000000
$baud = 9600

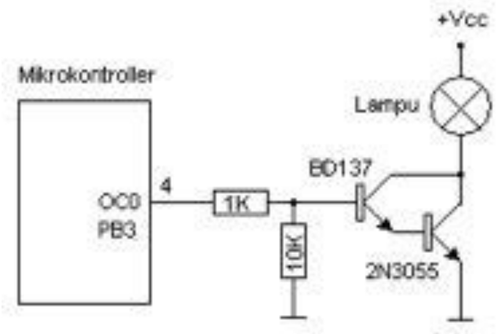
Dim Data_pb As Byte
Dim Pc As Byte

'Inisialisasi port
Config Portb = Output
Config Portc = Input

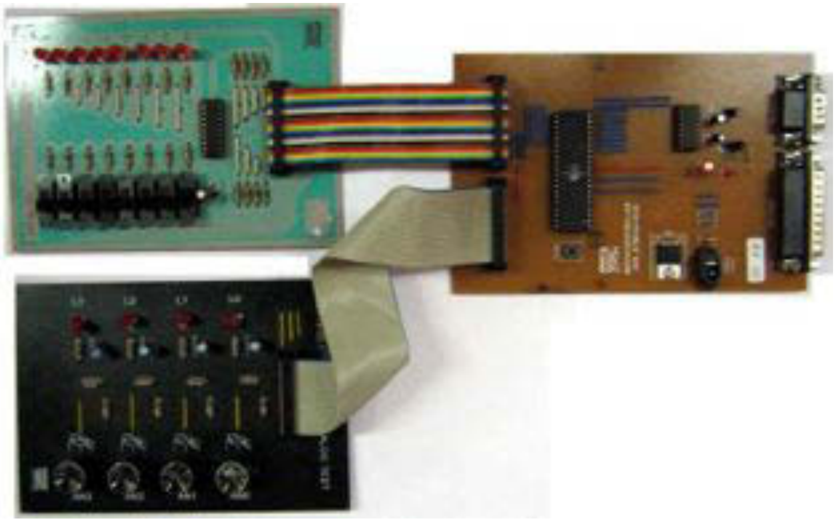
Do
   Input "Masukkan data PORTB : " , Data_pb
   Portb = Data_pb
   Pc = Pinc
   Print "Data PORTC = " , Pc
Loop
End
```



4.20 Pulse Wide Modulation (PWM)



Gambar 4.13 Rangkaian PWM dengan beban lampu LED



Gambar 4.14 Percobaan PWM

```
$regfile "m8535.dat"
$crystal = 4000000
Dim A0 As Word
Config Portb = Output
Config Adc = Single , Prescaler = Auto

Start Adc
Tccr0 = &B01101001
Ocr0 = 0

Do
   A0 = Getadc(0)
   A0 = A0 / 4
   Ocr0 = A0
Loop
End
```



Rangkuman

1. Sebelum mengakses port digital, harus dilakukan inisialisasi yaitu menentukan arah data sebagai masukan atau keluaran dengan instruksi **config.**

   Contoh :

   Config Portb = Output

   Config Portc = Input

   Untuk membaca port digital digunakan perintah “Pin”

   Contoh :

   A = PinA

2. Untuk program penyalaan LED diperlukan tunda waktu yaitu **Waitms**

   Contoh :

   Waitms 1000 ‘Tunda waktu 1 detik

   Waitms 100 ‘Tunda waktu 100 mili detik

3. Untuk membaca data analog harus melakukan inisialisasi ADC sebagai berikut :

   Config Adc = Single , Prescaler = Auto

   Untuk memulai pembacaan ADC dilakukan dengan instruksi :

   Start Adc

   Untuk mendapatkan hasil pembacaan ADC dilakukan dengan instruksi :

   W = Getadc(0)



4. Untuk menulis LCD harus melakukan inisialisasi pin LCD sebagai berikut :

   Config Lcdpin = Pin , Db4 = Portb.4 , Db5 = Portb.5 , Db6 = Portb.6 , Db7 = Portb.7 , E = Portb.3 , Rs = Portb.2

   Untuk menginisialisasi jenis LCD sesuai jumlah baris dan kolomdilakukan dengan instruksi :

   Config Lcd = 16 * 2

   Untuk menginisialisasi LCD supaya cursor tidak tampak dan tidak berkedip :

   Cursor Off Noblink

   Untuk membersihkan layer LCD digunakan instruksi **Cls**

   Untuk menulis LCD digunalkan instruksi **LCD** seperti contoh berikut :

   Upperline

   Lcd "A0: A2: "

   Lowerline

   Lcd "A1: A3: "

   Untuk menempatkan cursor pada posisi baris dan kolom tertentu digunakan instruksi **Locate**, Contoh : **Locate 2 , 13**

5. Komunikasi data serial dapat melalui USART menggunakan pin Tx dan Rx serta GND





KEGIATAN BELAJAR 5

Sebelum proses pembelajaran di kelas berlangsung, sebaiknya siswa mempersiapkan diri dengan belajar mandiri sesuai dengan urutan materi yang akan diberikan.Sebagai gambaran kegiatan belajar siswa seperti pada tabel berikut :

NO	KEGIATAN SISWA	KETERANGAN
1	Persiapan Kegiatan 1 1. Siswa membaca materi pendahuluan 2. Siswa mempelajari materi identifikasi arsitektur PLC 3. Siswa mempelajari hardware PLC dengan latihan menggambar pengawatan PLC 4. Siswa mencoba mengerjakan soal tes formatif 1	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 1
2	Persiapan Kegiatan 2 1. Siswa membaca materi pengertian program 2. Siswa mempelajari materi teknik pemrograman PLC 3. Siswa mempelajari instruksi sederhana pada program PLC 4. Siswa mencoba mengerjakan soal tes formatif 2	Kegiatan ini pada prinsipnya siswa belajar secara mandiri sebagai persiapan awal untuk menerima materi dari guru sesuai kegiatan 2
3	Persiapan Kegiatan 3 1. Siswa mempelajari materi transfer program ke dalam PLC dengan menggunakan software CX-programmer. 2. Siswa mempelajari pembuatan ladder diagram 3. Siswa mencoba mengerjakan soal tes formatif 3	Kegiatan ini pada prinsipnya siswa belajar secara mandiri untuk menerima materi dari guru sesuai kegiatan 3



Selanjutnya siswa mendengarkan penyampaian materi pembelajaran di setiap pertemuan oleh guru serta menyesuaikan dengan model pembelajran yang digunakan. Misalnya saatnya harus aktif mengerjakan soal maupun praktikum, maka siswa juga harus aktif dan kreatif. Melalui langkah kegiatan pembelajaran yang saling melengkapi diharapkan siswa dapat mencapai kompetensi yang distandarkan.

A. Tujuan Pembelajaran

Setelah mempelajari materi tentang arsitektur PLC, diharapkan siswa dapat:

1. mengidentifikasi arsitektur PLC
2. mengidentifikasi blok dalam PLC

B. Uraian Materi

- Dasar sistem kendali PLC, komponen dan spesifikasinya serta perbandingan sistem kendali PLC dengan sistem kendali yang lain.
- Teknik pemrograman PLC.
- Teknik pemasangan dan pengawatan peralatan input output.
- Penggunaan alat pemrogram dengan komputer yang dilengkapi dengan software ladder
- Pengoperasian sistem kendali PLC

C. Alokasi Waktu

4 jam pelajaran

D. Metode Pembelajaran

Teori dan Praktek

E. Media pembelajaran

- PC/Notebook
- Windows 7
- CX-Prgogrammer
- PLC Omron CMP1A



KEGIATAN 1

ARSITEKTUR PLC

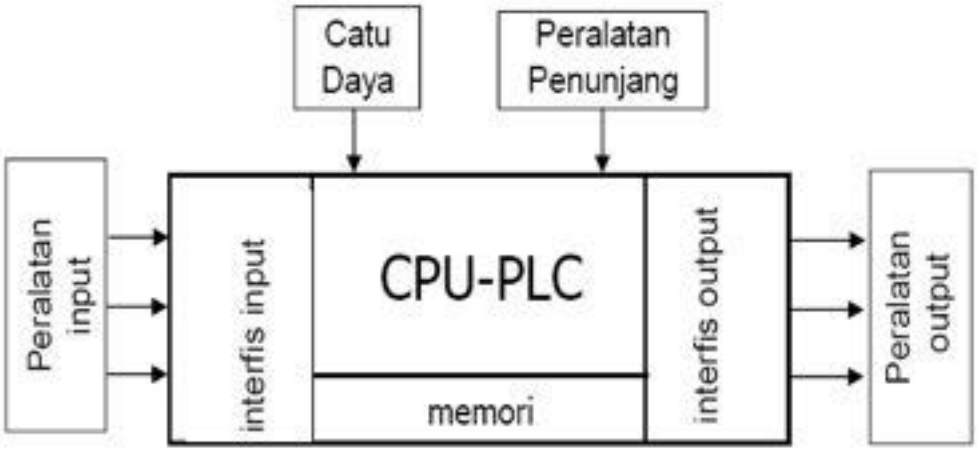
5.1 Pendahuluan

PLC (*Programmable Logic Controller*) adalah sebuah alat yang digunakan untuk menggantikan rangkaian sederetan relay yang dijumpai pada suatu *system control* secara konvensional.Dalam sistem otomasi, PLC merupakan 'jantung' sistem kendali. PLC dapat memonitor keadaan sistem melalui sinyal dari peralatan input, kemudian didasarkan atas logika program yang tersimpan di dalam memorinya, PLC akan menentukan rangkaian aksi pengendalian peralatan output luar.

PLC dapat digunakan untuk mengendalikan tugas-tugas sederhana yang berulang-ulang, atau di-interkoneksi dengan yang lain menggunakan komputer melalui sejenis jaringan komunikasi untuk mengintegrasikan pengendalian proses yang kompleks.

PLC bekerja secara kontak melalui sensor-sensor terkait (*input*) kemudian melakukan proses dan tindakan berupa menghidupkan atau mematikan keluaran (*output*) dengan logika 1 dan 0. Logika 1 mewakili ON, logika 0 mewakili OFF.

Cara kerja sistem kendali PLC dapat dipahami dengan diagram blok seperti ditunjukkan pada Gambar 1.



Gambar 1 Diagram blok PLC

Dari gambar terlihat bahwa PLC terdiri atas CPU (Central Processing Unit), memori, modul interface input dan output program kendali disimpan dalam memori program. Program PLC akan mengendalikan sinyal kontrol berdasar sinyal input yang diterima, berdasar alur program yang dibuat sehingga menghasilkan respon output sesuai yang diharapkan.

Penjelasan masing-masing komponen sebagai berikut:

a. CPU-PLC

CPU adalah mikroprosesor yang mengkordinasikan kerja sistem PLC. Mengeksekusi program, memproses sinyal input/output, dan mengkomunikasikan dengan peralatan luar. Memori akan menyimpan sistem operasi terprogram dan menyimpan data pemakai.

Ada dua jenis memori yaitu : ROM (*Read Only Memory*) dan RAM (*Random Access Memory*). ROM adalah memori yang hanya dapat diprogram sekali. Penyimpanan program dalam ROM bersifat permanen, maka ia digunakan untuk menyimpan sistem operasi. Ada sejenis ROM, yaitu EPROM (*Eraseable Programmable Read Only Memory*) yang isinya dapat dihapus dengan cara menyinari menggunakan sinar ultraviolet dan kemudian diisi program ulang menggunakan EPROM *Writer*.



*Interface*merupakan modul rangkaian yang digunakan untuk menyesuaikan sinyal pada peralatan luar. *Interface* input menyesuaikan sinyal dari peralatan input dengan sinyal yang dibutuhkan untuk operasi sistem. *Interface* output menyesuaikan sinyal dari PLC dengan sinyal untuk mengendalikan peralatan output.

b. Peralatan Input

Peralatan input merupakan sumber luar yang memberikan sinyal kepada PLC dan selanjutnya PLC memproses sinyal tersebut untuk mengendalikan peralatan output. Peralatan input itu antara lain:

- Berbagai jenis saklar, misalnya tombol, saklar toggle, saklar batas, saklar level, saklar tekan, saklar proximity.
- Berbagai jenis sensor, misalnya sensor cahaya, sensor suhu, sensor level,
- Rotary encoder dll.

c. Peralatan Output

Sistem otomasi tidak lengkap tanpa ada peralatan output yang dikendalikan. Peralatan output itu misalnya:

- Kontaktor
- Motor listrik
- Lampu
- Buzer

d. Peralatan Penunjang

Peralatan penunjang adalah peralatan yang digunakan dalam sistem kendali PLC, tetapi bukan merupakan bagian dari sistem secara nyata. Maksudnya, peralatan ini digunakan untuk keperluan tertentu yang tidak berkait dengan aktifitas pegendalian. Peralatan penunjang itu, antara lain :

- Berbagai jenis alat pemrogram, yaitu komputer/laptop, software ladder, konsol pemrogram, programmable terminal, dan sebagainya.



- Berbagai software ladder, yaitu: SSS, LSS, Syswin, dan CX Programmer.
- Berbagai jenis memori luar, yaitu: disket, CD ROM, flash disk.

e. Catu Daya

PLC adalah sebuah peralatan digital dan setiap peralatan digital membutuhkan catu daya DC. Catu daya ini dapat dicatu dari luar, atau dari dalam PLC itu sendiri. PLC tipe modular membutuhkan catu daya dari luar, sedangkan pada PLC tipe compact catu daya tersedia pada unit.

5.2 Pemilihan Unit Tipe PLC

Unit PLC dibuat dalam banyak model/ tipe. Pemilihan suatu tipe harusmempertimbangkan jenis catu daya, jumlah terminal input/ output, dan tipe rangkaian output.

a. Jenis Catu Daya

PLC adalah sebuah peralatan elektronik dan setiap peralatan elektronik untuk dapat beroperasi membutuhkan catu daya. Ada dua jenis catu daya untuk disambungkan ke PLC yaitu AC dan DC.

b. Jumlah I/O

Pertimbangan lain untuk memilih unit PLC adalah jumlah terminal I/O nya. Jumlah terminal I/O yang tersedia bergantung kepada merk PLC. Misalnya PLC merk OMRON pada satu unit tersedia terminal I/O sebanyak 10, 20, 30, 40 atau 60. Jumlah terminal I/O ini dapat dikembangkan dengan memasang Unit I/O Ekspansi sehingga dimungkinkan memiliki 100 I/O.

Pada umumnya, jumlah terminal input dan output megikuti perbandingan tertentu, yaitu 3 : 2. Jadi, PLC dengan terminal I/O sebanyak 10 memiliki terminal input 6 dan terminal output 4.

c. Tipe Rangkaian Output

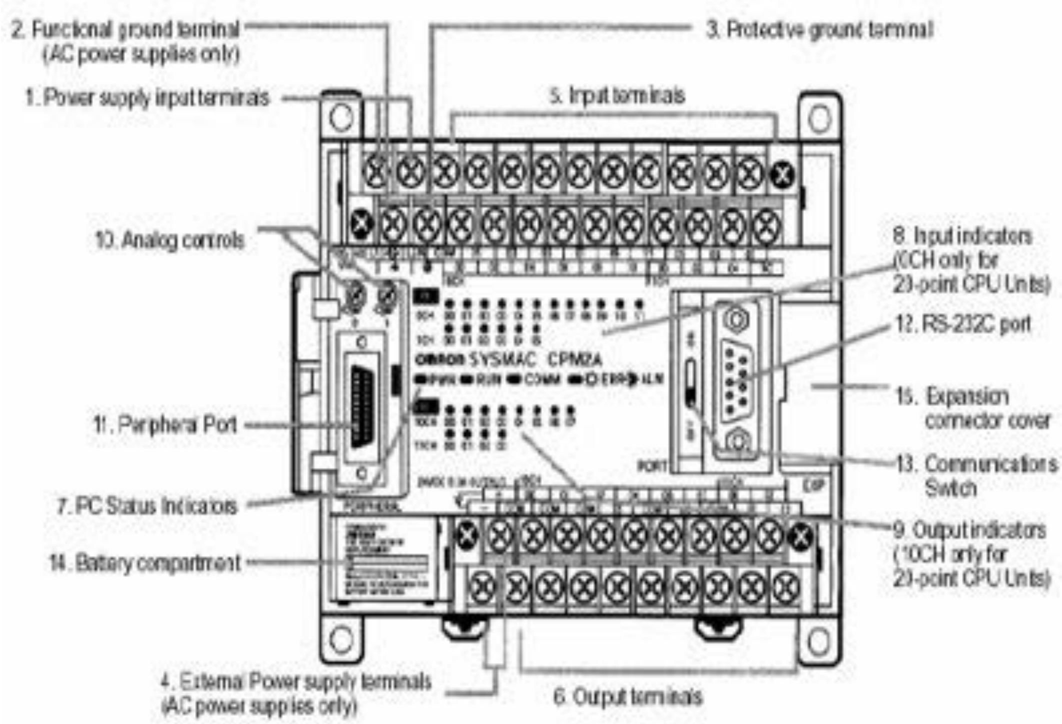
PLC dibuat untuk digunakan dalam berbagai rangkaian kendali. Bergantung kepada peralatan output yang dikendalikan, tersedia tiga tipe rangkaian output yaitu: output relai, output *transistor singking* dan output *transistor soucing.*



Jenis catu daya, jumlah I/O, dan tipe rangkaian output PLC OMRON CPM2A ditunjukkan pada tabel di bawah ini.

Number of I/O points	Power supply	Inputs	Outputs	Model
20 I/O points (12 inputs and 8 outputs)	100 to 240 VAC	24 VDC	Relay	CPM2A-20CDR-A
	24 VDC	24 VDC	Relay	CPM2A-20CDR-D
		24 VDC	Sinking Transistor	CPM2A-20CDT-D
		24 VDC	Sourcing Transistor	CPM2A-20CDT1-D
30 I/O points (18 inputs and 12 outputs)	100 to 240 VAC	24 VDC	Relay	CPM2A-30CDR-A
	24 VDC	24 VDC	Relay	CPM2A-30CDR-D
		24 VDC	Sinking Transistor	CPM2A-30CDT-D
		24 VDC	Sourcing Transistor	CPM2A-30CDT1-D
40 I/O points (24 inputs and 16 outputs)	100 to 240 VAC	24 VDC	Relay	CPM2A-40CDR-A
	24 VDC	24 VDC	Relay	CPM2A-40CDR-D
		24 VDC	Sinking Transistor	CPM2A-40CDT-D
		24 VDC	Sourcing Transistor	CPM2A-40CDT1-D
60 I/O points (36 inputs and 24 outputs)	100 to 240 VAC	24 VDC	Relay	CPM2A-60CDR-A
	24 VDC	24 VDC	Relay	CPM2A-60CDR-D
		24 VDC	Sinking Transistor	CPM2A-60CDT-D
		24 VDC	Sourcing Transistor	CPM2A-60CDT1-D

Komponen unit CPU PLC OMRON CPM2A ditunjukkan pada gambar berikut ini:



Gambar 3 CPU PLC OMRON CPM2A



5.3. Perbandingan Sistem Kendali Elektromagnet dan PLC

Pada sistem kendali relai elektromagnetik (kontaktor), semua pengawatan ditempatkan dalam sebuah panel kendali. Dalam beberapa kasus panel kendali terlalu besar sehingga memakan banyak ruang (tempat). Tiap sambungan dalam logika relai harus disambung. Jika pengawatan tidak sempurna, maka akan terjadi kesalahan sistem kendali.

Untuk melacak kesalahan ini, perlu waktu cukup lama. Pada umumnya, kontaktor memiliki jumlah kontak terbatas. Dan jika diperlukan modifikasi, mesin harus dimatikan. Jadi, panel kendali hanya cocok untuk proses yang sangat khusus. Ia tidak dapat dimodifikasi menjadi sistem yang baru dengan segera. Dengan kata lain, panel kendali elektromagnetik tidak fleksibel. Disamping itu biaya untuk pengadaan kontaktor dan relay jauh lebih mahal dibanding dengan pengadaan suatu sistem PLC.

Dari uraian di atas, dapat disimpulkan adanya kelemahan sistem kendali relai elektromagnetik sebagai berikut:

- Terlalu banyak pengawatan panel.
- Modifikasi sistem kendali sulit dilakukan.
- Pelacakan gangguan sistem kendali sulit dilakukan.
- Jika terjadi gangguan mesin harus diistirahatkan untuk melacak kesalahan sistem.
- Biaya pengadaan mahal.

Kesulitan-kesulitan di atas dapat diatasi dengan menggunakan sistemkendali PLC.



5.4. Keunggulan Sistem Kendali PLC

Keunggulan Sistem kendali PLC dibandingkan dengan sistem kendali elektromagnetik :

- Pengawatan sistem kendali PLC lebih ringkas.
- Modifikasi sistem kendali dapat dengan mudah dilakukan dengan cara mengganti progam kendali tanpa merubah pengawatan sejauh tidak ada tambahan peralatan input/output.
- Tidak diperlukan komponen kendali seperti timer dan hanya diperlukan sedikit kontaktor sebagai penghubung peralatan output ke sumber tenaga listrik.
- Kecepatan operasi sistem kendali PLC sangat cepat sehingga produktivitas meningkat.
- Biaya pembangunan sistem kendali PLC lebih murah dalam kasus fungsi kendalinya sangat rumit dan jumlah peralatan input/outputnya sangat banyak.
- Sistem kendali PLC lebih andal.
- Program kendali PLC dapat dicetak dengan cepat.

5.5. Penerapan Sistem Kendali PLC

Sistem kendali PLC digunakan secara luas dalam berbagai bidangantara lain untuk mengendalikan:

- Traffic light
- Lift
- Konveyor
- Sistem pengemasan barang
- Sistem perakitan peralatan elektronik
- Sistem pengamanan gedung
- Sistem pembangkitan tenaga listrik
- Robot
- Pemrosesan makanan



5.6. Langkah-Langkah Desain Sistem Kendali PLC

Pengendalian sistem kendali PLC harus dilakukan melalui langkah-langkah sistematik sebagai berikut:

a. Memilih PLC dengan spesifikasi yang sesuai dengan sistem kendali.

b. Memasang Sistem Komunikasi

c. Membuat program kendali

d. Mentransfer program ke dalam PLC

e. Memasang unit

f. Menyambung pengawatan I/O

g. Menguji coba program

h. Menjalankan program



Rangkuman

1. PLC adalah kependekan dari *Programmable Logic Controller* yang berarti pengendali yang bekerja secara logika dan dapat diprogram.

2. Peralatan sistem kendali PLC terdiri atas Unit PLC, memori, peralatan input, peralatan output, peralatan penunjang, dan catu daya.

3. Pemilihan suatu unit PLC didasarkan atas pertimbangan jenis catu daya untuk PLC, jumlah I/O dan tipe rangkaian output.

4.Penggunaan PLC harus memperhatikan spesifikasi teknisnya. Mengabaikan hal ini dapat mengakibatkan PLC rusak atau beroperasi secara tidak tepat (mal-function).

5. Dibandingkan sistem kendali elektromagnet, PLC lebih unggul dalam banyak hal, antara lain pengawatan sistem lebih sederhana, gambar sistem kendali mudah dicetak, lebih murah dalam kasus rangkaian kendali yang kompleks, dll.

6. PLC diterapkan dalam hampir segala lapangan industri sebagai pengendali mesin dan proses kerja alat.



Tes Formatif

1. Apakah yang dimaksud dengan sistem kendali?
2. Apakah perbedaan sistem kendali loop terbuka dan loop tertutup?
3. Apakah sesungguhnya PLC itu?
4. Sebutkan masing-masing tiga contoh:
   a. Alat input
   b. Alat output
   c. Alat penunjang
5. Gambarkan diagram blok yang menunjukkan hubungan masing-masingperalatan sistem kendali PLC !
6. Sebutkan lima keunggulan PLC dibandingkan dengan sistem kendali elektromagnet !
7. Jelaskan bahwa sistem kendali PLC lebih murah jika dibandingkan sistem kendali elektromagnet !
8. Sebutkan daerah penerapan PLC !



KEGIATAN 2

Teknik Pemrograman PLC

Tujuan Pembelajaran

Setelah selesai pembelajaran diharapkan siswa dapat :

- Merancang program kendali PLC sederhana
- Memasukkan program ke dalam PLC
- Mengecek kebenaran program

5.7Unsur-Unsur Program

Program kendali PLC terdiri atas tiga unsur yaitu : alamat, instruksi, dan operand. Alamat adalah nomor yang menunjukkan lokasi, instruksi, atau data dalam daerah memori. Instruksi harus disusun secara berurutan dan menempatkannya dalam alamat yang tepat sehingga seluruh instruksi dilaksanakan mulai dari alamat terendah hingga alamat tertinggi dalam program.

Instruksi adalah perintah yang harus dilaksanakan PLC. PLC hanya dapat melaksanakan instruksi yang ditulis menggunakan ejaan yang sesuai. Oleh karena itu, pembuat program harus memperhatikan tata cara penulisan instruksi.

Operand adalah nilai berupa angka yang ditetapkan sebagai data yang digunakan untuk suatu instruksi. Operand dapat dimasukkan sebagai konstanta yang menyatakan nilai angka nyata atau merupakan alamat data dalam memori.

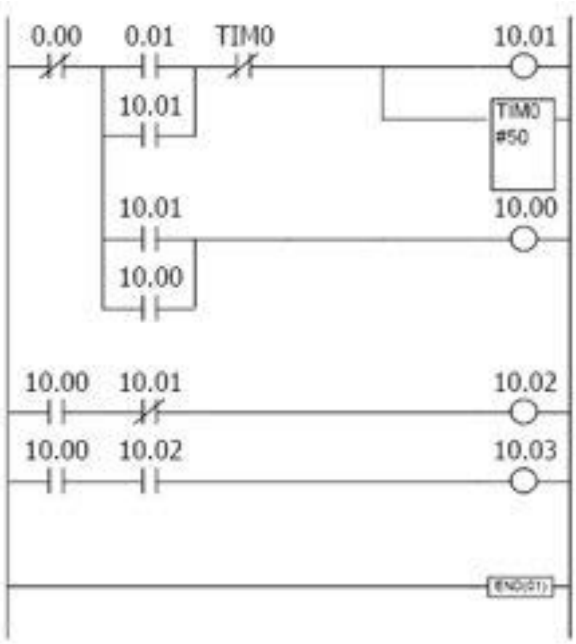


5.8 Bahasa Pemrograman

Program PLC dapat dibuat dengan menggunakan beberapa bahasa pemrograman. Bentuk program berbeda-beda sesuai dengan bahasa pemrograman yang digunakan. Bahasa pemrograman tersebut antara lain: diagram ladder, kode mneumonik, diagram blok fungsi, dan teks terstruktur. Beberapa merk PLC hanya mengembangkan program diagram ladder dan kode mneumonik.

a. Diagram Ladder

Digram ladder terdiri atas sebuah garis vertikal di sebelah kiri yang disebut *bus bar*, dengan garis bercabang ke kanan yang disebut *rung*. Sepanjang garis instruksi, ditempatkan kontak-kontak yang mengendalikan/mengkondisikan instruksi lain di sebelah kanan. Kombinasi logika kontak-kontak ini menentukan kapan dan bagaimana instruksi di sebelah kanan di- eksekusi. Contoh diagram ladder ditunjukkan pada gambar di bawah ini.



Gambar 1 Contoh Diagram Ladder



Terlihat dari gambar di atas bahwa garis instruksi dapat bercabang kemudian menyatu kembali. Sepasang garis vertikal disebut kontak (kondisi). Ada dua kontak, yaitu kontak NO (*Normally Open*) yang digambar tanpa garis diagonal dan kontak NC (*Normally Closed*) yang digambar dengan garis diagonal. Angka di atas kontak menunjukkan bit *operand*.

b. Kode *Mneumonik*

Kode *mneumonik* memberikan informasi yang sama persis seperti halnya diagram *ladder*. Sesungguhnya, program yang disimpan di dalam memori PLC dalam bentuk *mneumonik*, bahkan meskipun program dibuat dalam bentuk diagram *ladder*. Oleh karena itu, memahami kode *mneumonik* itu sangat penting. Berikut ini contoh program *mneumonik* :

Alamat	Instruksi	Operand
00000	LD	HR 01
00001	AND	0.01
00002	OR	0.02
00003	LD NOT	0.03
00004	OR	0.04
00005	AND LD	
00006	MOV(21)	
		0.00
		DM 00
00007	CMP(20)	
		DM 00
		HR 00

Gambar 2 Contoh program mneumonik

5.9 Struktur Daerah Memori

Program pada dasarnya adalah pemrosesan data dengan menggunakan berbagai instruksi di dalam memori PLC. Pemahaman area data dan pemahaman terhadap berbagai jenis instruksi merupakan hal yang sangat penting, karena dari segi inilah kita dapat membangun logika berpikir suatu pemrograman. Data yang merupakan *operand* suatu instruksi dialokasikan sesuai dengan jenis datanya. Tabel di bawah ini ditunjukkan daerah memori PLC CPM2A :



Daerah Data		Channel/ Words	Bit
IR	Daerah input	IR 000 s.d IR 009	IR 000.00 s.d IR 009.15
	Daerah output	IR 010 s.d IR 019	IR 010.00 s.d IR 019.15
	Daerah 'kerja'	IR 020 s.d IR 049 IR 200 s.d IR 227	IR 020.00 s.d IR 049.15 IR 200.00 s.d IR 227.15
SR		SR 228 s.d SR 255	SR 228.00 s.d SR 255.15
TR		---	TR 0 s.d TR 7
HR		HR 00 s.d HR 19	HR 00.00 s.d HR 19.15
AR		AR 00 s.d AR 23	AR 00.00 s.d AR 23.15
LR		LR 00 s.d LR 15	LR 00.00 s.d LR 15.15
TIM/ CNT		TC 000 s.d TC 255	

Gambar 3Peta memori address PLC

5.10 Instruksi Pemrograman

Terdapat banyak instruksi untuk memprogram PLC, tetapi tidak semua instruksi dapat digunakan pada semua model PLC.

a. Instruksi Diagram Ladder

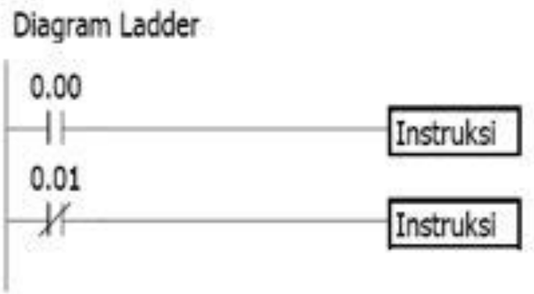
Instruksi diagram ladder adalah instruksi sisi kiri yang mengkondisikan instruksi lain di sisi kanan. Pada program diagram ladder instruksi ini disimbolkan dengan kontak-kontak seperti pada rangkaian kendali elektromagnet.

Instruksi diagram ladder terdiri atas enam instruksi ladder dan dua instruksi blok logika. Instruksi blok logika adalah instruksi yang digunakan untuk menghubungkan bagian yang lebih kompleks.

Instruksi *LOAD* dan *LOAD NOT*

Instruksi *LOAD* dan *LOAD NOT* menentukan kondisi eksekusi awal, oleh karena itu dalam diagram ladder disambung ke bus bar sisi kiri. Tiap instruksi memerlukan satu baris kode *mneumonik*.





Mneumonik

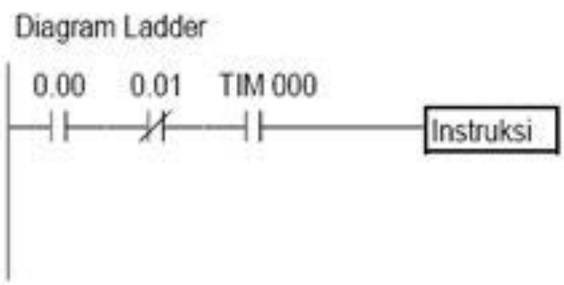
Alamat	Instruksi	Operand
00000	LD	0.00
00001	Instruksi	
00002	LD NOT	0.01
00003	Instruksi	

Gambar 4 Penggunaan Instruksi LOAD dan LOAD NOT

Jika misalnya hanya ada satu kontak seperti contoh di atas, kondisi eksekusi pada sisi kanan akan “AKTIF” jika kontaknya ON. Untuk instruksi LD yang kontaknya NO, kondisi eksekusinya akan ON jika IR 0.00 ON; dan untuk instruksi LD NOT yang kontaknya NC, akan ON jika IR 0.01 OFF.

Instruksi AND dan AND NOT

Jika dua atau lebih kontak disambung seri pada garis yang sama, kontak pertama berkait dengan instruksi LOAD atau LOAD NOT dan sisanya adalah instruksi AND atau AND NOT. Contah di bawah ini menunjukkan tiga kontak yang masing-masing menunjukkan instruksi LOAD, AND NOT, dan AND.



Mneumonik

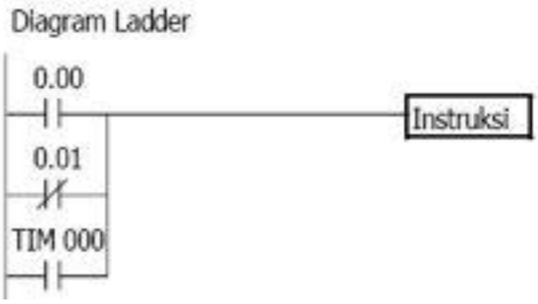
Alamat	Instruksi	Operand
00000	LD	0.00
00001	AND NOT	0.01
00002	AND	TIM 000
00003	Instruksi	

Gambar 5 Penggunaan Instruksi AND dan AND NOT



Instruksi OR dan OR NOT

Jika dua atau lebih kontak terletak pada dua instruksi terpisah dan disambung paralel, kontak pertama mewakili instruksi LOAD atau LOAD NOT dan sisanya mewakili instruksi OR atau OR NOT. Contoh berikut menunjukkan tiga kontak yang masing-masing mewakili instruksi LOAD, OR NOT, dan OR.



Mneumonik

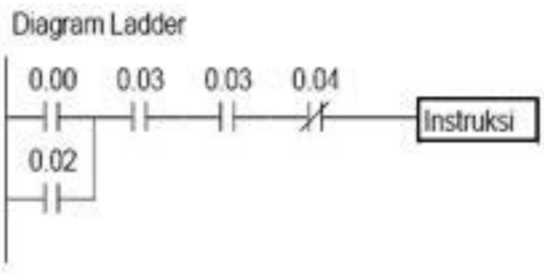
Alamat	Instruksi	Operand
00000	LD	0.00
00001	OR NOT	0.01
00002	OR	TIM 000
00003	Instruksi	

Gambar 6 Penggunaan Instruksi OR dan OR NOT

Instruksi akan mempunyai kondisi eksekusi ON jika salah satu di antara tiga kontak ON, yaitu saat IR 0.00 ON, saat IR 0.01 OFF, atau saat TIM 0.00 ON.

Kombinasi Instruksi AND dan OR

Jika instruksi AND dan OR dikombinasikan pada diagram yang lebih rumit, mereka dapat dipandang secara individual di mana tiap instruksi menampilkan operasi logika pada kondisi eksekusi dan status bit operand. Perhatikan contoh berikut ini hingga yakin bahwa kode mneumonik meliputi alur logika yang sama dengan diagram ladder.



Mneumonik

Alamat	Instruksi	Operand
00000	LD	0.00
00001	AND	0.01
00002	OR	0.02
00003	AND	0.03
00004	AND NOT	0.04
00005	Instruksi	

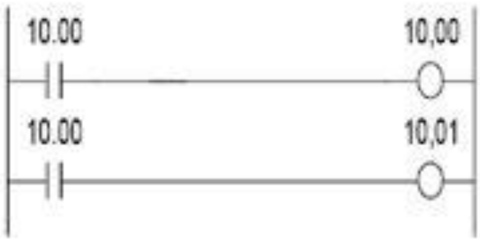
Gambar 7 Kombinasi Instruksi AND dan OR



Di sini AND terletak di antara statur IR 0.00 dan status IR 0.01 untuk menentukan kondisi eksekusi dengan meng-OR-kan status IR 0.02. Hasil operasi ini menentukan kondisi eksekusi dengan meng-AND-kan status IR 0.03 yang selanjutnya menentukan kondisi eksekusi dengan meng-AND-kan kebalikan status IR 0.04.

b. Instruksi OUT dan OUT NOT

Cara paling sederhana untuk meng-OUTPUT-kan kombinasi kondisi eksekusi adalah dengan meng-OUTPUT-kan langsung menggunakan instruksi OUTPUT dan OUTPUT NOT. Istruksi ini digunakan untuk mengendalikan status bit operand sesuai dengan kondisi eksekusi. Dengan instruksi OUTPUT, bit operand akan ON selama kondisi eksekusinya ON dan akan OFF selama kondisi eksekusinya OFF. Dengan instruksi OUTPUT NOT, bit operand akan ON selama kondisi eksekusinya OFF dan akan OFF selama kondisi eksekusinya ON.



Alamat	Instruksi	Operand
00000	LD	0,00
00001	OUT	10,00
00002	LD	0,01
00003	OUT NOT	10,01

Gambar 8 Penggunaan Instruksi OUTPUT dan OUTPUT NOT

Pada contoh di atas, IR 10.00 akan ON jika IR 0.00 ON dan IR 10.01 akan OFF selama IR 0.01 ON. Di sini IR 0.00 dan IR 0.01 merupakan bit input dan IR 10.00 dan IR 10.01 merupakan bit output yang ditetapkan untuk peralatan yang dikendalikan PLC.

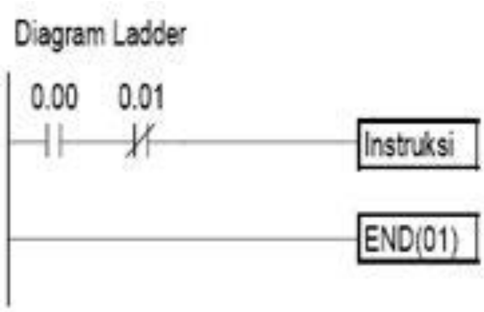
c. Instruksi END (01)

Instruksi terakihir yang diperlukan untuk melengkapi suatu program adalah instruksi END. Saat PLC menscan program, ia mengeksekusi semua instruksi hingga instruksi END pertama sebelum kembali ke awal program



dan memulai eksekusi lagi. Meskipun instruksi END dapat ditempatkan sembarang titik dalam program, tetapi intruksi setelah instruksi END pertama tidak akan diekseksekusi.

Nomor yang mengikuti instruksi END dalam kode mneumonik adalah kode fungsinya, yang digunakan saat memasukkan instruksi ke dalam PLC menggunakan konsol pemrogram. Instruksi END tidak memerlukan operand dan tidak boleh ada kontak ditempatkan pada garis instruksi yang sama. Jika dalam program tidak ada instruksi END, program tersebut tidak akan dieksekusi.



Mneumonik

Alamat	Instruksi	Operand
00000	LD	0.00
00001	AND NOT	0.01
00002	Instruksi	
00003	END(01)	

Gambar 9 Penggunaan Instruksi END(01)

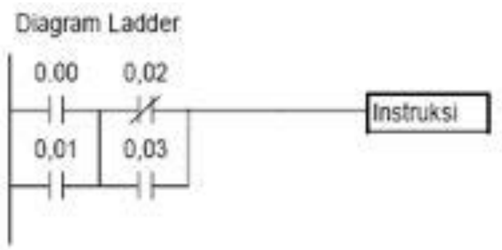
d. Instruksi Blok Logika

Jika rangkaian logika tidak dapat diwujudkan dengan instruksi AND, AND NOT, OR, atau OR NOT saja, maka perlu menggunakan instruksi blok logika. Perbedaannya adalah bahwa instruksi AND, AND NOT, OR, dan OR NOT **mengkombinasikan antar kondisi eksekusi dengan suatu bit operand**, sedangkan instruksi blok logika yang terdiri dari instruksi AND LOAD dan OR LOAD **mengkombinasikan kondisi eksekusi dengan kondisi eksekusi terakhir yang belum digunakan**. Instruksi blok logika tidak diperlukan dalam program diagram ladder, tetapi diperlukan hanya pada program mneumonik.



Instruksi AND LOAD

Instruksi AND LOAD meng-AND-kan kondisi eksekusi yang dihasilkan oleh dua blok logika.



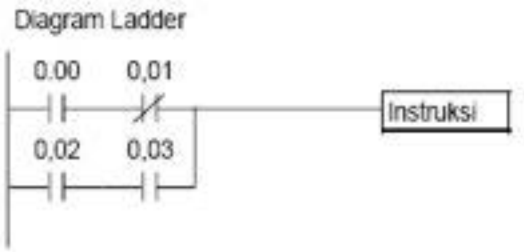
Mneumonik

Alamat	Instruksi	Operand
00000	LD	0.00
00001	OR	0.01
00002	LD NOT	0.02
00003	OR	0.03
00004	AND LD	
00005	Instruksi	

Gambar 10 Penggunaan Instruksi AND LOAD

e. Mengkode Instruksi Sisi Kanan Ganda

Jika terdapat lebih dari satu instruksi sisi kanan dengan kondisi eksekusi yang sama, masing-masing dikode secara berurutan mengikuti kondisi eksekusi terakhir pada garis instruksi. Pada contoh di bawah ini, garis instruksi terakhir berisi satu kontak lagi yang merupakan instruksi AND terhadap IR 0.03.



Mneumonik

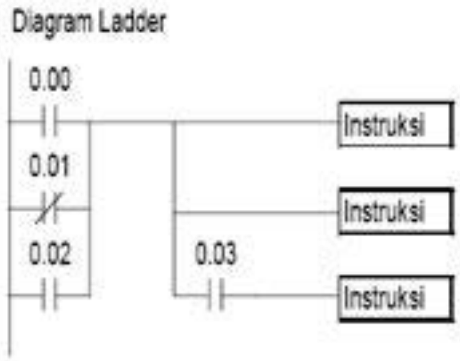
Alamat	Instruksi	Operand
00000	LD	0.00
00001	AND NOT	0.01
00002	LD	0.02
00003	ND	0.03
00004	OR LD	
00005	Instruksi	

Gambar 11 Penggunaan Instruksi OR LOAD

e. Mengkode Instruksi Sisi Kanan Ganda

Jika terdapat lebih dari satu instruksi sisi kanan dengan kondisi eksekusi yang sama, masing-masing dikode secara berurutan mengikuti kondisi eksekusi terakhir pada garis instruksi. Pada contoh di bawah ini, garis instruksi terakhir berisi satu kontak lagi yang merupakan instruksi AND terhadap IR 0.03.





Mneumonik

Alamat	Instruksi	Operand
00000	LD	0.00
00001	OR NOT	0.01
00002	OR	0.02
00003	Instruksi 1	
00004	Instruksi 2	
00005	AND	
00006	Instruksi 3	

Gambar 12 Mengkode Instruksi Sisi Kanan Ganda

f. Penggunaan Bit TR

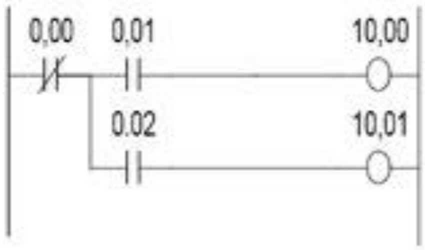
Bit TR (Temporarily Relay) digunakan untuk mempertahankan kondisi eksekusi pada garis instruksi bercabang. Hal ini dipertahankan karena garis instruksi dieksekusi menuju ke instruksi sisi kanan sebelum kembali ke titik cabang untuk mengeksekusi instruksi lainnya. Jika ada kontak pada garis instruksi setelah titik cabang, kondisi eksekusi untuk instruksi yang pertama tidak sama dengan kondisi pada titik cabang sehingga untuk mengeksekusi instruksi berikutnya menggunakan kondisi eksekusi titik cabang dan kontak lain setelah titik cabang tersebut.

Jika program dibuat dalam bentuk diagram ladder, tidak perlu memperhatikan bit TR karena bit TR hanya relevan pada pemrograman bentuk mneumonik.

Terdapat delapan bit TR, yaitu TR0 sampai dengan TR7 yang dapat digunakan untuk mempertahankan kondisi eksekusi sementara. Misalkan suatu bit TR ditempatkan pada suatu titik cabang, kondisi eksekusinya akan disimpan pada bit TR tersebut. Jika kembali ke titik cabang, bit TR mengembalikan kondisi eksekusi yang telah disimpan.



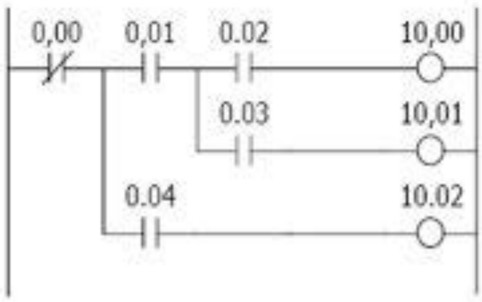
Penyimpanan kondisi eksekusi pada titik cabang menggunakan bit TR sebagai operand dari instruksi OUTPUT. Kondisi eksekusi ini kemudian dikembalikan setelah mengeksekusi instruksi sisi kanan dengan menggunakan bit TR yang sama sebagai operand dari instruksi LOAD.



Alamat	Instruksi	Operand
00000	LD NOT	0,00
00001	OUT	TR0
00002	AND	0.01
00003	OUT	10.00
00004	LD NOT	TR0
00005	AND	0.02
00006	OUT	10.01

Gambar 13 Penggunaan Bit TR

Contoh berikut ini menunjukkan penggunaan dua bit TR yaitu TR0 dan TR1 pada sebuah program.



Alamat	Instruksi	Operand
00000	LD NOT	0,00
00001	OUT	TR0
00002	AND	0.01
00003	OUT	TR1
00004	AND	0.02
00005	OUT	10.00
00006	LD NOT	TR1
00007	AND	0.03
00008	OUT	10.01
00009	LD NOT	TR0
00010	AND	0.04
00011	OUT	10.02

Gambar 14 Penggunaan Dua Bit TR



g. Instruksi Timer

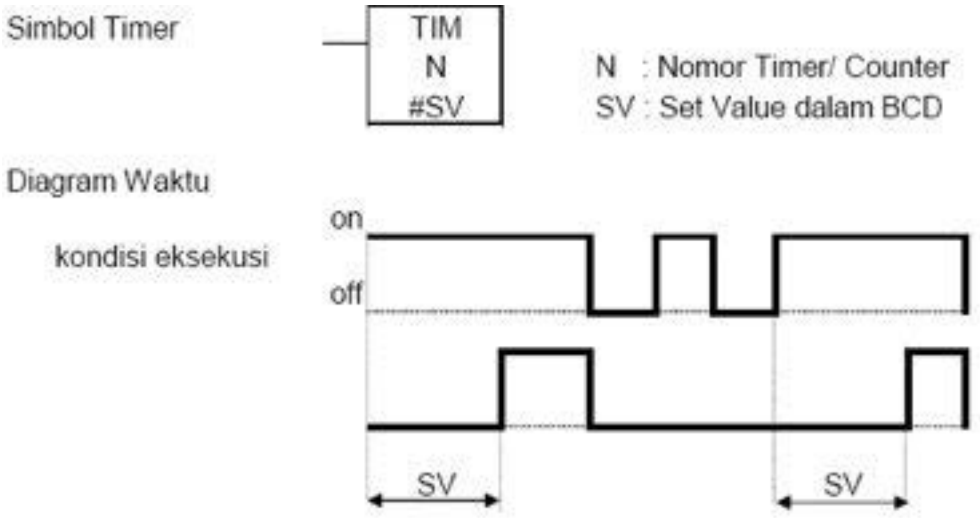
Instruksi Timer digunakan untuk operasi tunda waktu. Ia memerlukan dua operand yang terletak pada dua baris instruksi, yaitu baris pertama untuk nomor timer dan yug kedua untuk settig waktu (SV = Set Value). Meskipun demikian, instruksi Timer terletak dalam satu alamat.

Nomor Timer dipakai bersama untuk nomor Counter. Nomor Timer/ Counter hanya boleh digunakan sekali. Maksudnya, sekali nomor Timer/Counter telah digunakan, ia tidak boleh digunakan untuk instruksi Timer/Counter yang lain. Tetapi, nomor timer sebagai operand suatu kontak dapat digunakan sebanyak yang diperlukan.

Banyaknya nomor Timer/Counter bergantung kepada tipe PLC. Misalnya, PLC OMRON CPM1A, terdapat 128 nomor, yaitu dari 000 sampai dengan 127. tidak diperlukan awalan apapun untuk menyatakan nomor timer. Tetapi, jika nomor timer sebagai operand suatu kontak harus diberi awalan TIM.

SV dapat berupa konstanta atau alamat channel/words. Jika channel daerah IR sebagai unit input dimasukkan sebagai alamat channel, unit input ini harus disambung sedemikian sehingga SV dapat diset dari luar. Timer/ Counter yang disambung dengan cara ini hanya dapat diset dari luar dalam mode MONITOR atau RUN. Semua SV, termasuk yang diset dari luar harus dalam BCD (Binary Coded Decimal), yaitu bilangan desimal yang dikode biner. Penulisan SV harusdiawali dengan tanda #.





Gambar 16 Diagram Waktu Instruksi Timer

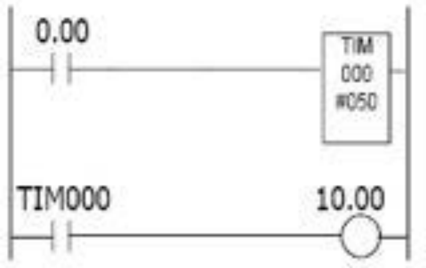
Timer bekerja saat kondisi eksekusinya beralih ke on dan direset (ke SV) saat kondisi eksekusinya beralih ke off. Jika kondisi eksekusi lebh lama daripada SV, completion flag, yaitu tanda yang menunjukkan hitungan waktu telah berakhir, tetap on hingga Timer direset. Timer akan reset jika trletak pada bagian program interlock saat kondisi eksekusi instruksi interlock (IL) off, dan saat terjadi pemutusan daya.

Jika dikehendaki timer tidak reset oleh dua keadaan tersebut, maka bit pulsa clock pada daerah SR untuk mencacah Counter yang menghasilkan Timer menggunakan instruksi Counter.

SV mempunyai harga antara 0000 sampai dengan 9999 (BCD) dalam satuan deci-detik. Jadi, misalnya menghendaki 10 detik, maka nilai SV harus 100. Jika SV dinyatakan tidak dalam BCD, akan muncul pesan kesalahan.



Di bawah ini diberikan program-program penerapan timer.

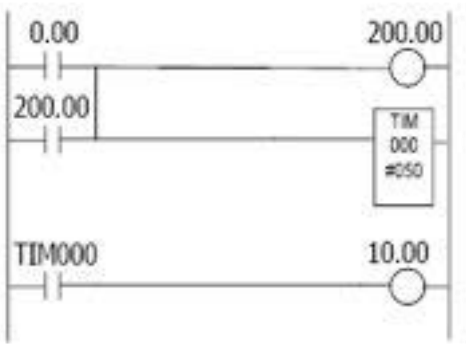


Alamat	Instruksi	Operand
00000	LD	0.00
00001	TIM	0.00
		#050
00002	LD	TIM 000
00003	OUT	10.00

Gambar 17 Contoh program Timer

Pada gambar di atas fungsi timer akan aktif jika kontak 0.00 = on. Lima detik kemudian (completion flag timer on) kontak TIM 000 akan aktif "ON"sehingga output 10.00 "ON". Jika lama kontak 0.00 on lebih pendek daripada SV, maka completion flag tetap off dan output 10.00 juga tetap off.

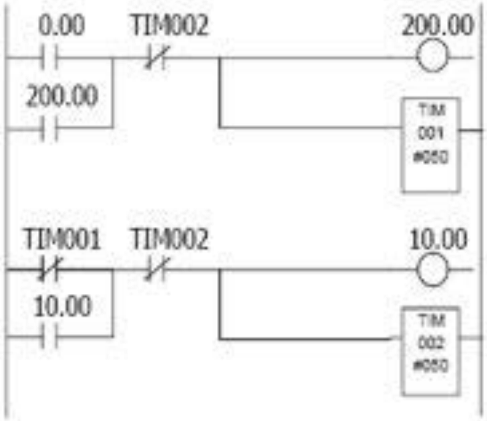
Agar dapat aktif meskipun kontak 0.00 hanya on sesaat, gunakan bit kerja untuk mengendalikan timer secara tidak langsung seperti ditunjukkan pada program berikut ini.



Alamat	Instruksi	Operand
00000	LD	0.00
00001	OR	200.00
00002	OUT	200.00
00003	TIM	000
		#050
00004	LD	TIM 000
00005	OUT	10.00

Gambar 18 Program Tunda On (2)





Alamat	Instruksi	Operand
00000	LD	0.00
00001	OR	200.00
00002	AND NOT	TIM 002
00003	OUT	200.00
00004	TIM	001
		#050
00005	LD NOT	TIM 001
00006	OR	10.00
00007	AND NOT	TIM 002
00008	OUT	10.00
00009	TIM	002
		#050

Gambar 19 Program Tunda On & Off

Untuk mengurangi kemungkinan terjadinya kesalahan dalam merancang program kendali, perlu diperhatikan hal-hal sebagai berikut :

1. Jumlah kondisi (kontak) yang digunakan seri atau paralel dan juga banyaknya perulangan penggunaan suatu bit tak terbatas sepanjang kapasitas memori PLC tidak dilampaui.
2. Diantara dua garis instruksi tidak boleh ada kondisi yang melintas secara vertikal.
3. Tiap garis instruksi harus memiliki sedikitnya satu kondisi yang menentukan eksekusi instruksi sisi kanan, kecuali untuk instruksi END(01), ILC(03) dan JME(05).
4. Dalam merancang diagram ladder harus memperhatikan kemungkinan instruksi yang diperlukan untuk memasukannya. Misalnya, pada gambar A di bawah ini diperlukan instruksi OR LOAD. Hal ini dapat dihindari dengan menggambar ulang diagram ladder seperti gambar B.



5.11 Langkah-langkah pembuatan program

Untuk membuat program kendali PLC ditempuh melalui langkah-langkah sistematis sebagi berikut :

1. Menguraikan urutan kendali

   Pembuatan program diawali dengan penguraian urutan kendali. Ini dapat dibuat dengan menggunakan kalimat-kalimat logika, gambar-gambar, diagram waktu, atau bagan alir (flow chart).

2. Menetapkan bit operand untuk peralatan input/ output.

   Bit operand untuk peralatan input/ output mengacu pada daerah memori PLC yang digunakan. Bit operand dapat dipilih secara bebas sejauh berada pada jangkah daerah memori yang dalokasikan. Tetapi, penggunaan secara bebas sering menjadikan ketidak-konsistenan sehingga menjadikan program kendali keliru. Oleh sebab itulah penggunaan bit operand harus ditetapkan sebelum program dibuat. Inventarisir semua peralatan input dan output yang akan disambung ke PLC, kemudian tetapkan bit operandnya.

   Jumlah bit oprand yang tersedia bergantung kepada tipe PLC yang dispesifikasikan menurut jumlah input-outputnya. Perbandingan jumlah bit input dan output pada umumnya 3 : 2. Misalnya PLC dengan I/O 10 memiliki bit input sejumlah 6 dan bit output 4. Di bawah ini diberikan contoh daerah memori PLC OMRON CPM1A-10CDRA.

Daerah Data		**Words**	**Bit**
IR (Internal Relay)	Input	0	0.00 – 0.11
	Output	10	10.00 – 10.07
	Kerja (internal)	200 – 231	200.00 – 231.15
TR (Temporarilly Relay)			TR0 – TR7
Timer/counter		TC0 – TC7	



3. Membuat program kendali

   Program kendali PLC dapat dibuat dengan diagram ladder atau kode mneumonik. Pemilihan tipe program sesuai dengan jenis alat pemrogram yang akan digunakan untuk memasukkan program ke dalam PLC. Jika diguinakan komputer pilihlah diagram ladder dan jika digunakan konsol pemrogram gunakan kode mneumonik.

5.12 Program Kendali Motor

Terdapat berbagai macam operasi motor induksi, suatu motor yang paling banyak digunakan sebagai penggerak mesin industri. Tetapi, hanya ada beberapa prinsip operasi motor induksi yaitu :

- Operasi motor satu arah putaran
- Operasi motor dua arah putaran
- Operasi motor dua kecepatan
- Operasi motor start bintang segitiga
- Operasi beberapa motor kendali kerja berurutan

5.13 Program Kendali Motor Satu arah Putaran

1. Urutan Kendali Motor

   Jika tombol Start ditekan, motor berputar searah jarum jam, dan jika kemudian tombol Start dilepaskan[1)], motor tetap berputar dalam arah yang sama. Jika tombol Stop ditekan, motor berhenti berputar.

2. Penetapan Bit I/O

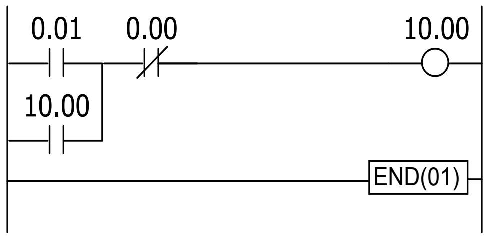
No	Alat input/output	Bit operand	Fungsi
1	Tombol Stop	0.00	Menghentikan operasi motor
2	Tombol Start	0.01	Menjalankan motor
3	Kontaktor[2)]	10.00	Menghubungkan motor ke jaringan



Keterangan :

1. Kecuali untuk operasi yang sangat khusus, secara umum operasi menjalankan motor adalah dengan menekan tombol Start dan jika kemudian tombol ini dilepas motor akan tetap berputar. Maka, selanjutnya untuk menjalankan motor cukup disebutkan dengan menekan tombol Start saja.
2. Motor berdaya kecil dapat disambung langsung ke PLC. Tetapi, untuk motor berdaya cukup dengan arus nominal diatas kemampuan PLC harus menggunakan kontaktor sebagai penghubung motor ke jaringan.
3. Program Kendali Motor Satu Arah Putaran

Diagram Ladder



Mneumonik

Alamat	Instruksi	Operand
00000	LD	0.01
00001	OR	10.00
00002	AND NOT	0.00
00003	END(01)	

Gambar 20 Program Kendali Motor Satu Arah Putaran

5.14 Program Kendali Motor Dua Arah Putaran

1. Urutan Kendali Motor

   Jika tombol Forward (FWD) ditekan, motor berputar searah jarum jam dan jika yang ditekan tombol Reverse (REV), motor berputar berlawanan arah jarum jam. Tombol STOP digunakan untuk menghentikan operasi motor setia saat.

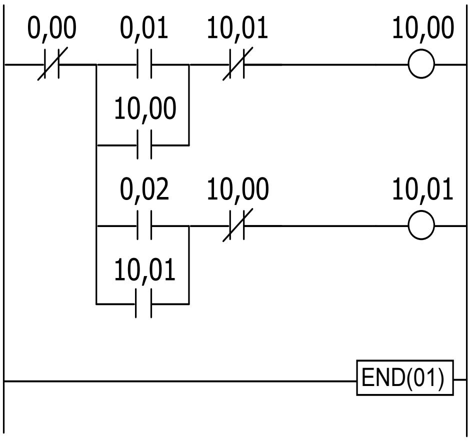


2. Penetapan Bit I/O

No	Alat input/output	Bit operand	Fungsi
1	Tombol Stop	0.00	Menghentikan operasi motor
2	Tombol Fwd	0.01	Menjalankan motor searah jarum jam
3	Tombol Rev	0.02	Menjalankan motor berlawanan arh jarum jam
4	Kontaktor K1	10.00	Kontaktor putaran searah jarum jam
5	Kontaktor K2	10.01	Kontaktor putaran berlawanan arh jarum jam

3. Program Kendali PLC

Diagram Ladder



Mneumonik

Alamat	Instruksi	Operand
00000	LD NOT	0,00
00001	OUT	TR0
00002	LD	0,01
00003	OR	10,00
00004	AND LD	
00005	AND NOT	10,01
00006	OUT	10
00007	LD	TR0
00008	LD	0,02
00009	OR	10,01
00010	AND LD	
00011	AND NOT	10,00
00012	OUT	10,01
00013	END(01)	

Gambar 21 Program Kendali Motor Dua Arah Putaran



5.15 Program Kendali Motor Sistem Start Bintang Segitiga

1. Urutan Kendali Motor

   Jika tombol Start ditekan, motor berputar dalam sambungan bintang. Lima detik kemudian, motor berputar dalam sambungan segitiga. Tombol Stop untuk menghentikan operasi motor setiap saat.

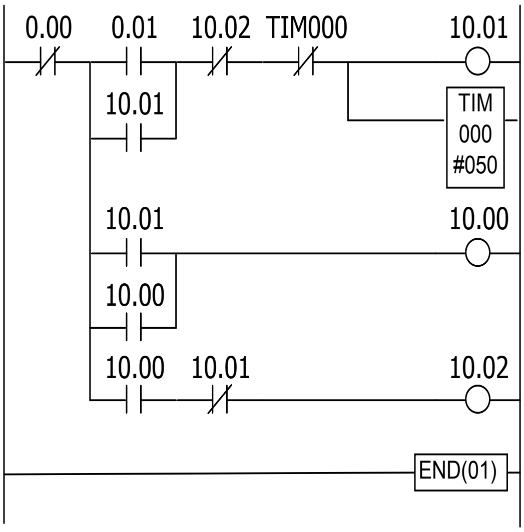
2. Penetapan Bit I/O

No	Alat input/output	Bit operand	Fungsi
1	Tombol Stop	0.00	Menghentikan operasi motor
2	Tombol Start	0.01	Menjalankan motor
3	Kontaktor K1	10.00	Kontaktor utama
4	Kontaktor K2	10.01	Kontaktor bintang
5	Kontaktor K3	10.02	Kontaktor segitiga

3. Program Kendali PLC



Diagram Ladder



Mneumonik

Alamat	Instruksi	Operand
00000	LD NOT	0.00
00001	OUT	TR0
00002	LD	0.01
00003	OR	10.01
00004	AND LD	
00005	AND NOT	10.02
00006	AND NOT	TIM000
00007		#050
00008	OUT	10.01
00009	LD	TR0
00010	LD	10.01
00011	OR	10.00
00012	AND LD	
00013	OUT	10.00
00014	LD	TR0
00015	AND	10.00
00016	AND NOT	10.01
00017	OUT	10.02
00018	END(01)	

Rangkuman

1. Program kendali PLC terdiri atas tiga unsur yaitu alamat, instruksi dan operand.
2. Program PLC dapat dibuat dengan diagram ladder atau kode mneumonik. Pemilihan tipe program ditentukan oleh alat pemrogram yang akan digunakan.
3. Untuk dapat membuat program kendali PLC, pemrogram harus memahami struktur daerah memori PLC yang akan digunakan. Daerah memori PLC berbeda-beda sesuai dengan tipe PLC.
4. Memahami instruksi pemrograman memegang peranan paling penting dalam pembuatan program kendali. Terdeapat banyak sekali instruksi pemrograman, tetapi tidak semua instruksi dapat duterapkan pada semua tipe PLC.



5. Setiap program selalu diawali dengan instruksi LOAD dan diakhiri dengan instruksi END. Tanpa instruksi END program tidak dapat dieksekusi.
6. Program dieksekusi dengan menscan mulai dari alamat terendah hingga ke alamat tertinggi yaitu instruksi END. Pada diagram ladder ini berarti program dikesekusi mulai dari atas ke bawah bila garis instruksi bercabang, dan kemudian ke kanan hingga mengeksekusi instruksi sisi kanan.
7. Pembuatan program PLC harus dilakukan secara sistematis, yaitu mendeskripsikan sistem kendali, menetapkan operand untuk alat input/ output, baru membuat program.
8. Banyak sekali variasi program kendali motor sebagai penggerak mesin. Tetapi, untuk operasi motor induksi, suatu motor yang paling banyak digunakan sebagai penggerak mesin, secara prinsip hanya ada beberapa operasi motor yaitu operasi motor satu arah putaran, operasi dua arah putaran, operasi dua kecepatan, operasi dengan start bintang segitiga, operasi berurutan dan operasi bergantian.

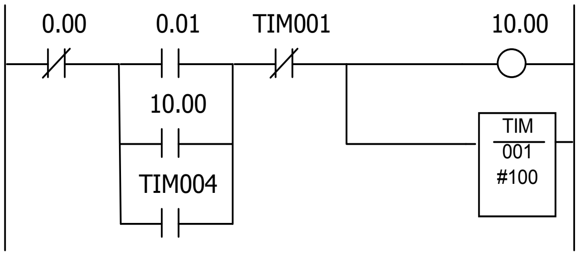


Tes Formatif

1. Apa yang dimaksud dengan program ?
2. Sebutkan dua macam bentuk program kendali PLC !
3. Sebutkan unsur-unsur sebuah program !
4. Sebutkan enam macam instruksi diagram ladder !
5. Bilamana bit TR digunakan dalam pembuatan program?
6. Instruksi manakah yang digunakan untuk operasi penundaan waktu ?
7. Apa yang dimaksud dengan SV (Set Value) ?
8. Sebutkan contoh instruksi yang tidak memerlukan operand !
9. Sebutkan contoh instruksi yang tidak memerlukan kondisi !
10. Mengapa bit operand untuk perlatan I/O harus ditetapkan terlebih dahulu sebelum membuat diagram ladder ?
11. Konversikan program diagram ladder berikut ini menjadi program mneumonik !

Alamat	Instruksi	Operand
00000	LD NOT	0.00
00001	OUT	TR 0
00002	AND LD	0.01
00003	OUT	10.00
00004	LD NOT	TR 0
00005	AND	0.02
00006	OUT	10.01

15. Konversikan program mneumonik berikut ini menjadi program diagram ladder !





KEGIATAN 3

TRANSFER PROGRAM KE DALAM PLC

A. Tujuan Pembelajaran

Setelah pemelajaran siswa dapat menggunakan software CX-Programmer untuk:

a. Membuat program diagram ladder
b. Mentransfer program ke dalam PLC

B. Uraian Materi

1. Mode operasi PLC
2. Konfigurasi hardware transfer program ke PLC
3. Memprogram menggunakan CX-Programmer

C. Alokasi Waktu

4 jam pelajaran

D. Metode Pembelajaran

Teori dan Praktek

E. Media pembelajaran

- PC/Notebook
- Windows 7
- CX Programmer



MEMASUKKAN PROGRAM KE DALAM PLC

5.16 Mode Operasi PLC

Operasi PLC dikategorikan dalam tiga mode yaitu: PROGRAM, MONITOR, dan RUN. Pilihan mode operasi harus dipilih dengan tepat sesuai dengan aktifitas dalam sistem kendali PLC.

Mode PROGRAM digunakan untuk membuat dan mengedit program, menghapus memori, atau mengecek kesalahan program. Pada mode ini, program tidak dapat dieksekusi/ dijalankan.

Mode MONITOR digunakan menguji operasi sistem, seperti memonitor status operasi, melaksanakan instruksi *force set* dan *force reset* bit I/O, merubah SV (Set Value) dan PV (Present Value) timer dan counter, merubah data kata, dan mengedit program online.

Mode RUN digunakan untuk menjalankan program. Status operasi PLC dapat dimonitor dari peralatan pemrogram, tetapi bit tdk dapat di paksa set/ reset dan SV/PV timer dan counter tidak dapat diubah.

5.17Konfigurasi hardware transfer program ke PLC

1. Alat Pemrogram

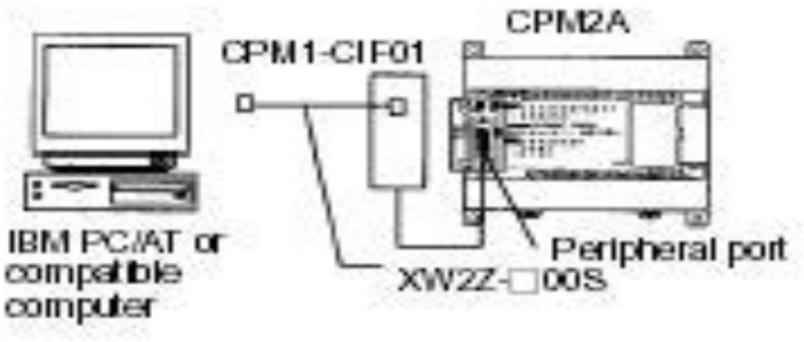
Ada beberapa jenis alat untuk memasukkan program ke dalam PLC yaitu komputer yang dilengkapi dengan software ladder misalnya CX-Programmer, Konsol Pemrogram, dan Programmable Terminal. Dengan software ladder CX-Programmer, program yang dimasukkan ke dalam PLC dapat berbentuk diagram ladder atau kode mneumonik..

2. Sambungan Alat Pemrogram

PLC dapat disambung ke Konsol Pemrogram atau komputer dengan software ladder seperti CX-Programmer, SSS (Sysmac Support Software) atau Syswin, dan Programmable Terminal.

Komunikasi Host Link adalah komunikasi antara PLC dan komputer yang didalamnya diinstal software ladder. Komputer dapat disambung ke port peripheral atau port RS-232C PLC. Port peripheral dapat beroperasi dalam mode Host Link atau mode peripheral bus. Port RS-232C beroperasi hanya dalam mode Host Link

Komputer dapat disambung ke port peripheral PLC dengan adapter RS-232C : CQM1-CIF02 atau CPM1-CIF01.



Gambar 2 Sambungan komunikasi Host Link

3. Memasukkan Program Menggunakan CX-Programmer

CX Programmer adalah software ladder untuk PLC merk OMRON. Ia beroperasi di bawah sistem operasi Windows, oleh sebab itu pemakai software ini diharapkan sudah familier dengan sistem operasi Windows antara lain untuk menjalankan software program aplikasi, membuat file, menyimpan file, mencetak file, menutup file, membuka file, dan keluar dari (menutup) software program.

Ada beberapa persyaratan minimum yang harus dipenuhi untuk bisa mengoperasikan CX Programmer secara optimal yaitu:

- Komputer IBM PC/AT kompatibel
- CPU Pentium I minimal 133 MHz
- RAM 32 Mega bytes



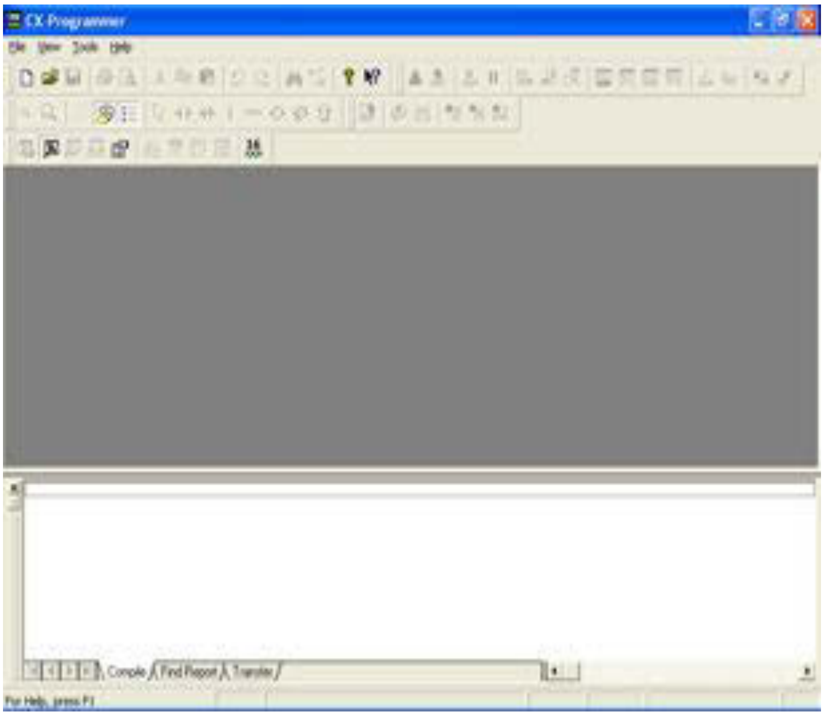
- Hard disk dengan ruang kosong kurang lebih 100 MB
- Monitor SVGA dengan resolusi 800 x 600

5.18 Memprogram menggunakan CX-Programmer

a. Menjalankan CX Programmer

Ada banyak cara untuk menjalankan suatu software termasuk CX Programmer. Berikut ini ditunjukkan cara umum menjalankan software dalam sistem operasi Windows.

Klik tombol Start > Program > OMRON > CX-Programmer > CX-Programmer. Akan tampil Layar CX Programmer sebagai berikut:



Gambar 4 Layar interface utama

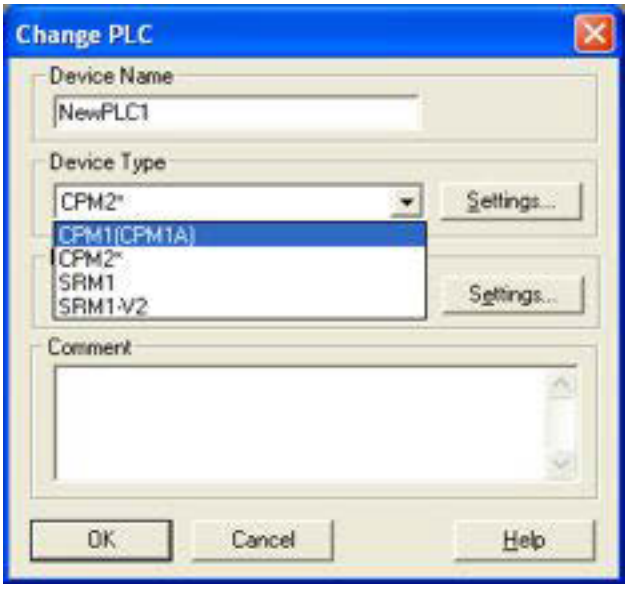
Ada beberapa menu/command yang perlu diketahui pada layar CX-Programmer utama yaitu:



Menu/Command	Fungsi
File>New	Membuat file baru
File>Open	Membuka file
File>Exit	Keluar dari CX-Programmer
View>Toolbar	Menampilkan/ menyembunyikan toolbar
Tool>Option	Mengatur beberapa opsi :
Help Topic	Meminta penjelasan menurut topik
Help Content	Meminta penjelasan menurut isi

b. Membuat file baru

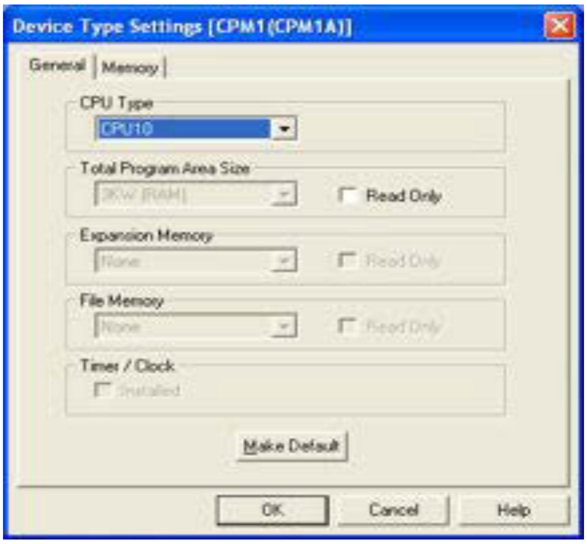
Klik File, **New** untuk membuat file baru.Kotak dialog **Change PLC** ditampilkan



Gambar 5. Kotak dialog merubah PLC

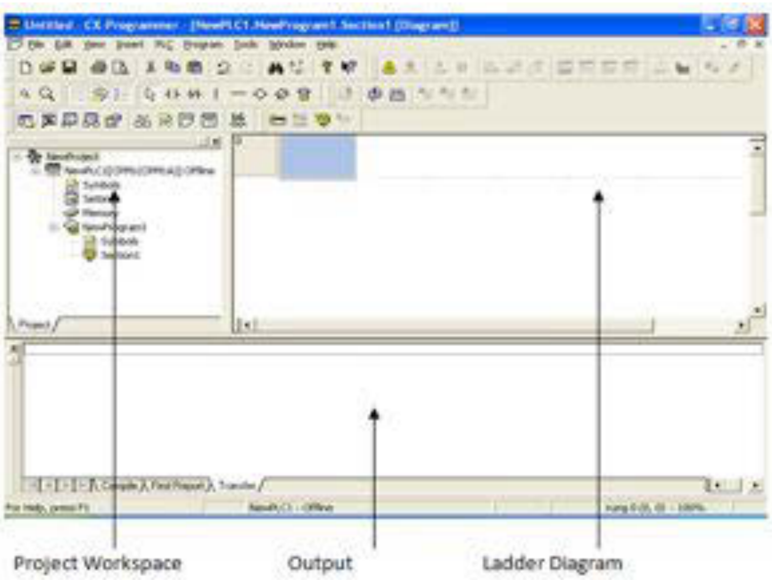
Pada kotak **Device Type**, klik tanda ▼ untuk memilih tipe PLC yang akan digunakan. Kemudian klik **Setting** untuk memilih jumlah input/output PLC. Kotak dialog Device Type Setting ditampilkan.





Gambar 6. Kotak dialog Device Type Setting

Pada General, CPU Type, klik tand ▼ untuk memilih jumlah I/O PLC, **OK.** Kembali ke kotak dialog Change PLC, pilih **OK.** Layar CX-Programmer ditampilkan.



Gambar 7. Layar CX-Programmer



Secara default ada tiga window tampil secara bersamaan, yaitu:

1) Window diagram ladder

Tempat untuk mengerjakan (menggambar) diagram ladder.

2) Window Project Workspace

Window Project Workspace (Ruang Kerja Proyek) menampilkan proyek sebagai struktur hierarkhi antara PLC dan rincian program. Penjelasan beberapa obyek dalam struktur ini sebagai berikut:

- **PLC** Menampilkan dan merubah tipe PLC, menampilkan mode operasi PLC
- **Symbols Global** Menampilkan simbol global, yaitu simbol yang digunakan secara umum untuk semua program. Yang dimaksud symbols adalah operand dalam daerah memori PLC.
- **Program** Menampilkan nama program (proyek)
- **Symbol Local** Menampilkan simbol lokal, yaitu simbol yang digunakan hanya pada program yang sedang aktif.
- **Section** Menampilkan/ menyembunyikan tampilan diagram ladder.

3) Window Output

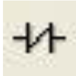
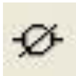
Window output akan menampilkan kesalahan dalam menulis diagram ladder. Kesalahan juga ditunjukkan secara langsung dalam window diagram ladder, dimana akan muncul tampilan warna merah pada bagian program yang salah.

1. Menggambar Diagram Ladder

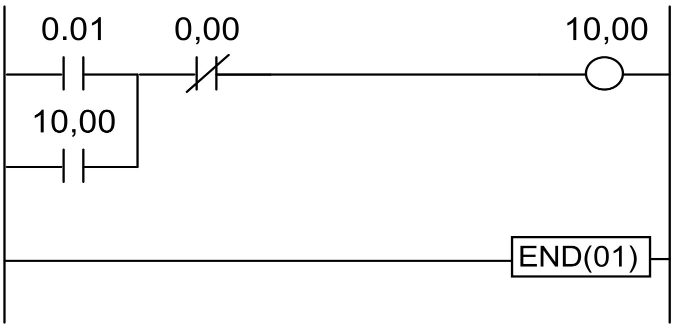
CX-Programmer membebaskan pemakai untuk membuat program dalam bentuk diagram ladder atau mneumonik. Tetapi, akan lebih baik menggunakan program diagram ladder.

Pemakai juga dibebaskan untuk menggunakan operasi toolbar, atau shortcut keyboard. Fungsi masing-masing toolbar dan shortcut ditunjukkan pada tabel berikut ini :



Menu/ Comand	**Toolbar**	**Shortcut**
Insert>Contact>Normally Open		C
Insert>Contact>Normally Closed		/
Insert>Vertical>Up		U
Insert>Vertical>Down		V
Insert>Horizontal		-
Insert>Coil>Normally Open		O
Insert>Coil>Normally Closed		Q
Insert>Instruction		I

Misalnya, program ladder di bawah ini akan dibuat menggunakan CX-Programmer !

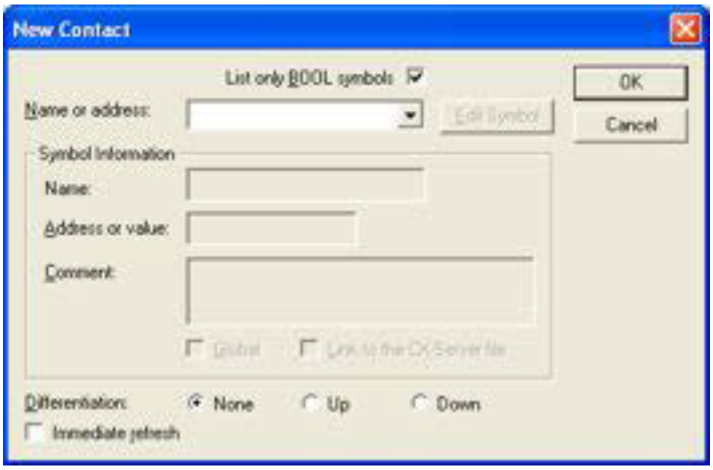


Gambar 8. Program Diagram ladder

Lakukan prosedur persiapan hingga tampil layar CX-Programmer seperti dijelaskan diatas.

1) Tempatkan kursor pada sel kiri atas. Klik **Insert > Contact > Normally Open** atau , maka muncul kotak dialog New Contact





Gambar 9. Kotak dialog New Contact

Pada kotak Name or address, ketik ‘1’ untuk menulis operand 0.01. Klik **OK** atau tekan **Enter**. Kursor akan bergeser ke kanan satu sel.

2) Klik **Insert > Contact > Normally Closed** atau , ketik ‘0’ untuk menulis operand 0.00, Klik **OK** atau tekan **Enter.**

3) Klik **Insert > Coil > Normally Open** atau , maka muncul kotak dialog New Coil :



Gambar 10. Kotak dialog New Coil

4) Ketik ‘1000’ untuk menulis operand 10.00. Klik **OK** atau tekan **Enter.**

5) Tekan **Enter**, untuk menambah baris pada rung yang sama. Kursor berpindah ke awal baris baru.

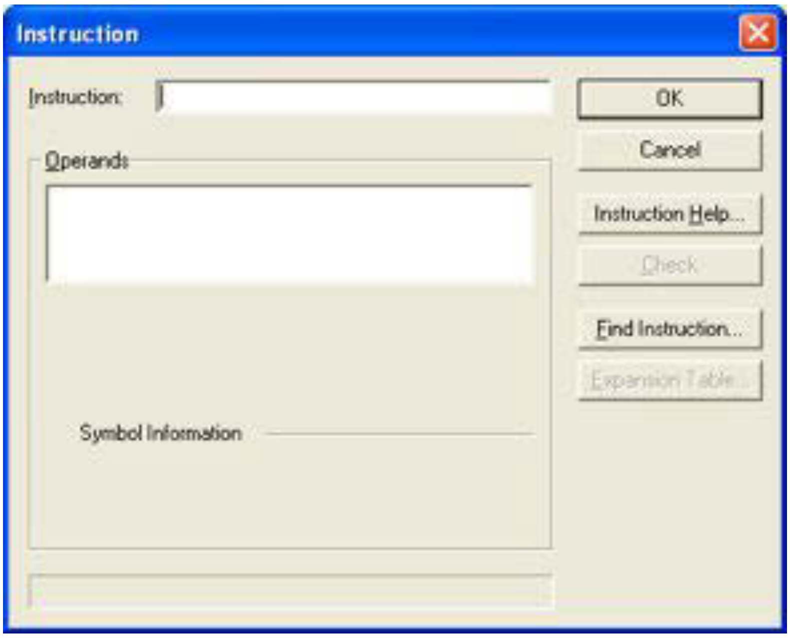
6) Klik **Insert > Contact > Normally Open** atau , ketik ‘1000’, **OK**.



7) Klik **Insert > Vertical > Up** atau diantara kontak NO 0.01 dan kontak NC 0.00.

8) Tekan tombol **Esc** untuk menon-aktifkan toolbar yang sedang aktif. Pindahkan kursor ke awal rung baru dengan menggunakan tombol anak panah. Begitu kursor berpindah ke rung baru, diagram ladder secara otomatis mengembang ke kanan.

9) Klik **Insert > Instruction**  untuk menulis instruksi lainnya. Muncul kotak dialog Instruction sebagai berikut:



Gambar 11. Kotak dialog Instruction

Ketik END pada kotak Instruction, **OK**. Pindahkan kursor ke rung baru. Seperti tadi, instruksi END mengembang ke kanan otomatis.

2. Menyimpan File

1. Klik **File Save** atau  untuk menyimpan file. Muncul kotak dialog Save CX-Programmer File.

2. Klik  pada kotak Save input untuk memilih tempat memori dimana file akan disimpan. Misalkan file akan disimpan di floppy disk, maka



pilih 3½ Floppy (A:).Pada kotak File Name, tulis nama file, misalnya 'M1A'.

Pada kotak Save input type, klik untuk memilih tipe file. Pilih CX-Programmer Project Files, lalu klik Save. Sekarang, file proyek telah disimpan dalam memori dan file ini dapat diakses setiap saat untuk ditindak-lanjuti.

3. **Menutup File**

Klik **File> close** untuk menutup file.

4. **Membuka file proyek**

1. Klik **File>Open** atau untuk membuka file yang pernah dibuat. Klik pada kotak Save input tempat dimana file disimpan.

2. Klik pada kotak file name untuk memilih nama-nama file yang ada pada memori.

3. Klik pada kotak file of type untuk memilih tipe file, lalu klik Open, maka file yang dipilih akan dibuka.

5. **Mentransfer program ke dalam PLC**

Operasi pemrograman PLC dibedakan menjadi operasi offline dan operasi online. Operasi offline adalah kegiatan pemrograman yang tidak memerlukan unit PLC, misalnya membuat diagram ladder, menyimpan file. Operasi online adalah kegiatan pemrograman yang tidak dapat dilakukan tanpa adanya unit PLC, misalnya mentransfer program, memonitor program, dan menjalankan program.

Transfer program dibedakan menjadi dua yaitu: **Download** dan **Upload**. Download adalah pemindahan program dari komputer ke PLC, sedangkan upload adalah pemindahan program dari PLC ke komputer.



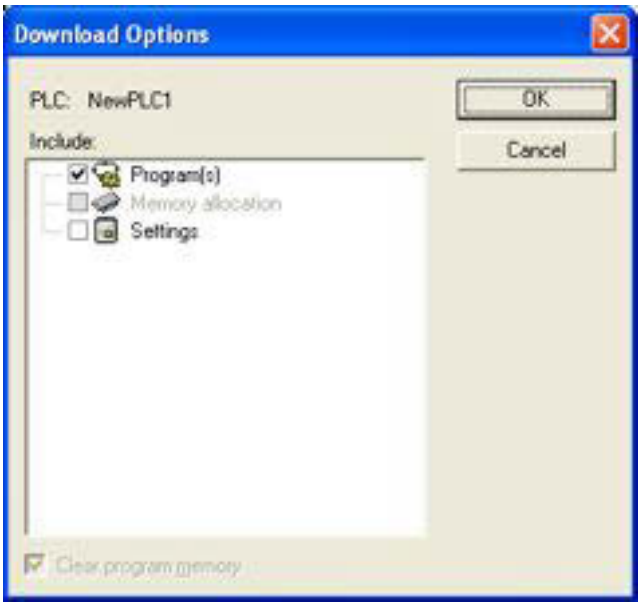
Operasi transfer program hanya dapat dilakukan dalam mode operasi PROGRAM. Jika PLC tidak dalam mode ini, CX-Programmer akan merubah mode secara otomatis.

Prosedur transfer program dari komputer ke PLC (Download) sebagai berikut :

1) Klik menu **PLC > Work Online**, untuk beralih ke operasi online. Pada layar muncul pesan meminta konfirmasi untuk beralih ke operasi online.

   Klik **Yes** untuk melanjutkan operasi. Latar belakang layar diagram ladder berubah menjadi gelap yang menunjukkan anda sedang berada pada operasi on-line.

2) Klik menu **PLC > Transfer > To PLC** untuk mendown-load program. Muncul kotak dialog yang meminta penjelasan apa saja yang akan di transfer: program atau setting, atau keduanya. Setelah dipilih, klik **OK.**



Gambar 12. Download option



Kotak dialog konfirmasi transfer program ditampilkan. Konfirmasi ini penting karena perintah transfer program akan berpengaruh terhadap PLC yang disambung.

3) Klik **Yes** untuk melanjutkan operasi. Pada layar ditunjukkan operasi transfer program sedang berlangsung. Jika selesai, ada informasi: **Download successful**.Klik "**OK"**Program anda sekarang sudah ada di PLC.



Rangkuman

1. Ada tiga mode operasi PLC yaitu mode PROGRAM, MONITOR, dan RUN. Mode PROGRAM digunakan untuk membuat dan mengedit program, menghapus memori, atau mengecek kesalahan program. Mode MONITOR digunakan menguji operasi sistem. Mode RUN digunakan untuk menjalankan program.
2. Ada beberapa jenis alat pemrogram antara lain CX-Programmer, Konsol Pemrogram, dan Programmable Terminal.
3. Dengan software ladder CX-Programmer, program yang dimasukkan ke dalam PLC dapat berbentuk diagram ladder atau kode mneumonik, tetapi Konsol Pemrogram hanya dapat memasukkan program dalam bentuk kode mneumonik.
4. Memasukkan program ke dalam PLC menggunakan CX-Programmer melalui prosedur membuat diagram ladder, baru mentransfer program.



Tes Formatif

1. Sebutkan tiga jenis alat yang digunakan untuk memprogram PLC!
2. Apakah perbedaan utama antara pemrograman PLC menggunakan software ladder dan menggunakan Konsol Pemrogram?
3. Sebutkan software komputer untuk memprogram PLC merk OMRON!
4. Apakah yang dimaksud dengan komunikasi Host Link?
5. Sebutkan lima syarat komputer untuk dapat digunakan mengoperasikan software CX-Programmer secara optimal!
6. Sebutkan perintah standar dalam CX Programmer untuk:
   a. Mengambar kontak NO
   b. Menggambar kontak NC
   c. Menggambar garis horisontal
   d. Menggambar garis vertikal ke bawah
   e. Menggambar garis vertikal ke atas
   f. Menggambar kumparan
   g. Menggambar instruksi END
   h. Beralih dari operasi offline ke online
   i. Merubah mode operasi PLC
   j. Mentransfer program dari komputer ke PLC
7. Apakah syarat-syarat untuk dapat mentransfer program dari komputer ke dalam PLC?
8. Apakah yang dimaksud dengan download?
9. Indikator apakah yang menunjukkan bahwa operasi transfer program telah berhasil?



Daftar Pustaka

Andrianto, Heri. “*Pemrograman Mikrokontroller AVR ATMega 16 Menggunakan bahasa C (Code Vision)”,* Bandung, Informatika.

Putra, Agfianto Eko. “*Tip dan Trik Mikrokontroller AT89 dan AVR Tingkat Pemula Hingga Lanjut”.* Gaya Media. Yogyakarta,2010.

Putranto, Agus. “*Memprogram Peralatan Sistem Otomasi Elektronik yang Berkaitan dengan I/O berbantuan Mikrokontroller ATmega8535* ”., Malang, 2008.

Putranto, A. “*Teknik Otomasi Industri Untuk Sekolah Menengah Kejuruan*”. Direktorat Pembinaan Sekolah Menengah dan Kejuruan. Jakarta, 2008.

http://www.vbtutor.net/index.php/visual-basic-2010-tutorial/